aws

Guide de l’utilisateur

Amazon EKS

Amazon EKS: Guide de l’utilisateur

Table of Contents

Présentation d'Amazon EKS

Amazon Elastic Kubernetes Service (Amazon EKS) est un service géré qui élimine le besoin d'installer, d'exploiter et de gérer votre propre plan de contrôle Kubernetes sur Amazon Web Services (AWS). Kubernetes est un système open source qui automatise la gestion, la mise à l'échelle et le déploiement des applications conteneurisées.

Fonctions d'Amazon EKS

Voici les principales caractéristiques d'Amazon EKS :

Mise en réseau et authentification sécurisées

Amazon EKS intègre vos Kubernetes charges de travail aux services AWS réseau et de sécurité. Il s'intègre également à AWS Identity and Access Management (IAM) pour fournir une authentification à vos Kubernetes clusters.

Mise à l'échelle facile des clusters

Amazon EKS vous permet d'augmenter ou de diminuer facilement vos clusters Kubernetes en fonction de la demande de vos charges de travail. Amazon EKS prend en charge la scalabilité automatique horizontale du Podenfonction du processeur ou de métriques personnalisées, et la scalabilité automatique du cluster en fonction de la demande de l'ensemble de la charge de travail.

Expérience Kubernetes gérée

Vous pouvez apporter des modifications à vos Kubernetesclusters en utilisant`eksctl`, AWS Management Console, AWS Command Line Interface (AWS CLI), l'API, `kubectl` etTerraform.

Haute disponibilité

Amazon EKS fournit une haute disponibilité pour votre plan de contrôle dans plusieurs zones de disponibilité.

Intégration avec les AWS services

Amazon EKS s'intègre à d'autres services AWS, fournissant une plate-forme complète pour le déploiement et la gestion de vos applications conteneurisées. Vous pouvez également résoudre plus facilement les problèmes de vos charges de travail Kubernetes avec divers outils d'observabilité.

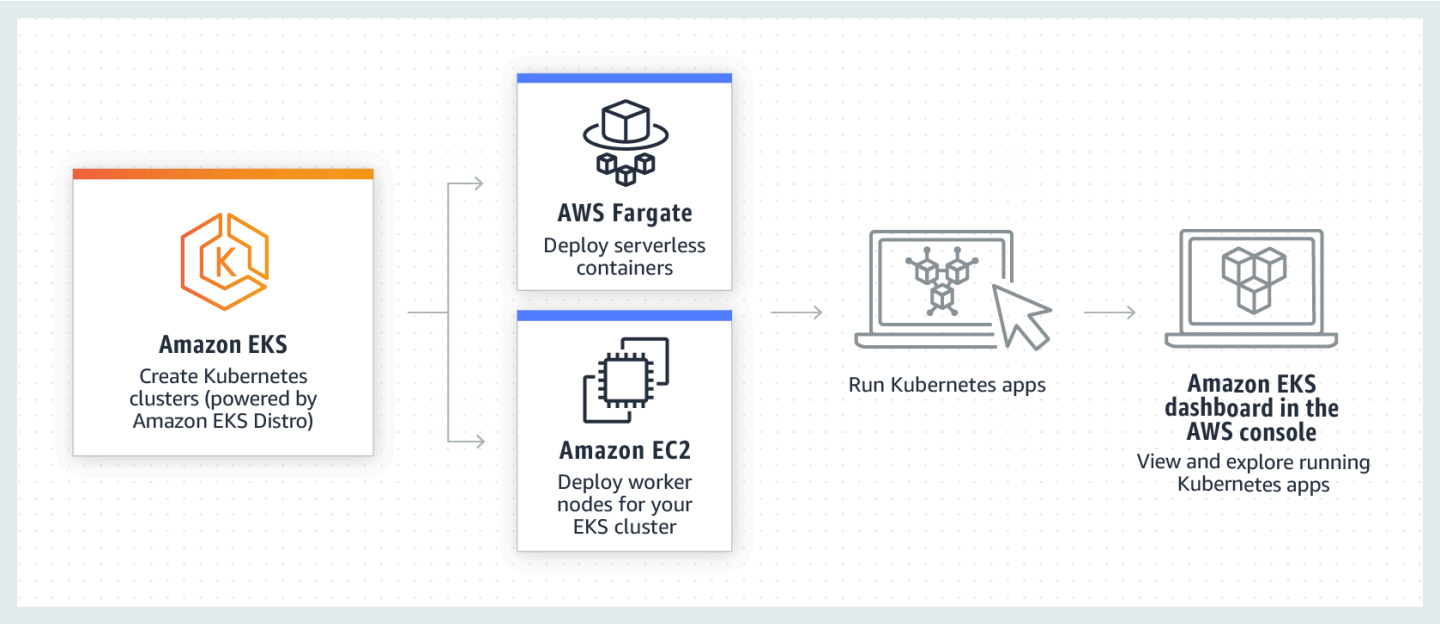
Pour en savoir plus sur les autres fonctionnalités d'Amazon EKS, consultez la section Fonctionnalités d'Amazon EKS.

Mise en route avec Amazon EKS

Pour créer votre premier cluster et ses ressources associées, consultez Démarrer avec Amazon EKS. En général, la mise en route avec Amazon EKS implique les étapes suivantes.

1. Création d'un cluster : commencez par créer votre cluster à l'aide de `eksctl` AWS Management Console, AWS CLI, ou de l'un des AWS SDK.
2. Choisissez votre approche en matière de ressources informatiques : choisissez entre AWS FargateKarpenter les groupes de nœuds gérés et les nœuds autogérés.
3. Configuration – Configurez les contrôleurs, les pilotes et les services nécessaires.
4. Déployer des charges de travail – Personnalisez vos charges de travail Kubernetes pour utiliser au mieux les ressources et les capacités du type de nœud que vous avez choisi.
5. Gestion – Supervisez vos charges de travail en intégrant des services AWS pour rationaliser les opérations et améliorer les performances des charges de travail. Vous pouvez consulter les informations relatives à vos charges de travail à l'aide du AWS Management Console.

Le diagramme suivant montre le flux de base de l'exécution d'Amazon EKS dans le cloud. Pour en savoir plus sur les autres options de déploiement Kubernetes, consultez Options de déploiement.



Tarification pour Amazon EKS

Un cluster Amazon EKS se compose d'un plan de contrôle et du calcul Amazon Elastic Compute Cloud (Amazon EC2) ou Fargate sur lequel vous exécutez des Pods. Pour plus d'informations sur la tarification du plan de contrôle, consultez Tarification Amazon EKS. Amazon EC2 et Fargate fournissent tous les deux :

On-Demand instances

Payez les instances que vous utilisez à la seconde, sans engagement à long terme ou paiement initial. Pour plus d'informations, consultez Tarification à la demande Amazon EC2 et Tarification AWS Fargate.

, Savings Plans

Vous pouvez réduire vos coûts en vous engageant pour une utilisation continue, en USD par heure, pour une durée de 1 à 3 ans. Pour plus d'informations, consultez Tarification avec Savings Plans.

Cas d'utilisation courants dans Amazon EKS

Amazon EKS propose de robustes services gérés par Kubernetes surAWS, conçus pour optimiser les applications conteneurisées. Voici quelques-uns des cas d'utilisation les plus courants d'Amazon EKS, qui vous aideront à tirer parti de ses atouts pour répondre à vos besoins spécifiques.

Déploiement d'applications à haut niveau de disponibilité

Elastic Load Balancing vous permet de vous assurer que vos applications sont hautement disponibles sur plusieurs Zones de disponibilité.

Création d'architectures de microservices

Utilisez des fonctionnalités de découverte de service Kubernetes avec AWS Cloud Map ou Amazon VPC Lattice pour créer des systèmes résilients.

Automatisation du processus de publication de logiciel

Gérez les pipelines d'intégration et de déploiement continus (CICD) qui simplifient le processus de création, de test et de déploiement automatisés des applications.

Exécution d'applications sans serveur

Utilisez AWS Fargate avec Amazon EKS pour exécuter des applications sans serveur. Cela signifie que vous pouvez vous concentrer uniquement sur le développement d'applications, pendant qu'Amazon EKS et Fargate gèrent l'infrastructure sous-jacente.

Exécution de charges de travail de machine learning

Amazon EKS est compatible avec les cadres de machine learning les plus courants tels que TensorFlow, MXNet et PyTorch. Grâce à la prise en charge GPU, vous pouvez gérer efficacement des tâches de machine learning même complexes.

Déploiement cohérent sur site et dans le cloud

Utilisez Amazon EKS Anywhere pour exploiter des clusters Kubernetes sur votre propre infrastructure à l'aide d'outils compatibles avec Amazon EKS dans le cloud.

Exécution d'un traitement par lots et de charges de travail big data rentables

Utilisez des instances Spot pour exécuter votre traitement par lots et vos charges de travail big data, telles que Apache Hadoop et Spark, à une fraction du prix. Cela vous permet de profiter de la capacité Amazon EC2 inutilisée à prix réduits.

Sécurisation des applications et garantie de conformité

Mettez en œuvre des pratiques de sécurité strictes et maintenez la conformité avec Amazon EKS, qui s'intègre à des services de sécurité AWS tels que AWS Identity and Access Management(IAM), Amazon Virtual Private Cloud (Amazon VPC) et AWS Key Management Service (AWS KMS). Cela garantit la confidentialité et la protection des données conformément aux normes du secteur.

Architecture Amazon EKS

Amazon EKS s'aligne sur l'architecture générale du cluster de Kubernetes. Pour plus d'informations, consultez Composants Kubernetes dans la documentation Kubernetes. Les sections suivantes résument certains détails supplémentaires relatifs à l'architecture d'Amazon EKS.

Plan de contrôle

Amazon EKS garantit que chaque cluster possède son propre plan de contrôle Kubernetes. Cette conception permet de séparer l'infrastructure de chaque cluster, sans aucun chevauchement entre les clusters ou comptes AWS. La configuration inclut :

Composants distribués

Le plan de contrôle positionne au moins deux instances de serveur API et trois instances `etcd` qui s'exécutent sur trois zones de disponibilité AWS dans une Région AWS .

Performances optimales

Amazon EKS surveille et ajuste activement les instances du plan de contrôle afin de maintenir des performances optimales.

Résilience

Si une instance de plan de contrôle échoue, Amazon EKS la remplace rapidement, en utilisant une autre zone de disponibilité si nécessaire.

Disponibilité constante

En exécutant des clusters sur plusieurs zones de disponibilité, uncontrat de niveau de service (SLA) pour la disponibilité des points de terminaison du serveur API fiable est atteint.

Amazon EKS utilise Amazon Virtual Private Cloud (Amazon VPC) pour limiter le trafic entre les composants du plan de contrôle au sein d'un même cluster. Les composants du cluster ne peuvent pas afficher ou recevoir des communications provenant d'autres clusters ou d'autres comptes AWS, sauf en cas d'autorisation avec des stratégies de contrôle d'accès basé sur les rôles (RBAC) Kubernetes.

Calcul

Outre le plan de contrôle, un cluster Amazon EKS possède un ensemble de machines de travail appelées nœuds. La sélection du type de nœud de cluster Amazon EKS approprié est essentielle pour répondre à vos besoins spécifiques et optimiser l'utilisation des ressources. Amazon EKS propose les types de nœuds primaires suivants :

AWS Fargate

Fargateest un moteur de calcul sans serveur pour les conteneurs qui élimine le besoin de gérer les instances sous-jacentes. Avec Fargate, vous spécifiez les besoins en ressources de votre application, et AWS provisionne, met à l'échelle et gère automatiquement l'infrastructure. Cette option est idéale pour les utilisateurs qui privilégient la facilité d'utilisation et souhaitent se concentrer sur le développement et le déploiement d'applications plutôt que sur la gestion de l'infrastructure.

Karpenter

Karpenter est un outil Cluster Autoscaler de Kubernetes flexible hautes performances qui permet d'améliorer la disponibilité des applications et l'efficacité du cluster. Karpenter lance des ressources de calcul de taille appropriée en réponse à l'évolution de la charge de l'application. Cette option permet de provisionner des ressources de calcul juste à temps qui répondent aux exigences de votre charge de travail.

Groupes de nœuds gérés

Les groupes de nœuds gérés sont un mélange d'automatisation et de personnalisation pour gérer un ensemble d'instances Amazon EC2 au sein d'un cluster Amazon EKS. AWS prend en charge des tâches telles que l'application de correctifs, la mise à jour et la mise à l'échelle des nœuds, simplifiant ainsi les aspects opérationnels. En parallèle, les arguments `kubelet` personnalisés sont pris en charge, ce qui ouvre la voie à des politiques avancées de gestion du processeur et de la mémoire. De plus, ils renforcent la sécurité grâce à des rôles AWS Identity and Access Management (IAM) pour les comptes de service, tout en réduisant le besoin d'autorisations distinctes par cluster.

Nœuds autogérés

Les nœuds autogérés offrent un contrôle total sur vos instances Amazon EC2 au sein d'un cluster Amazon EKS. Vous êtes chargé de gérer, de mettre à l'échelle et de maintenir les nœuds, ce qui vous donne un contrôle total sur l'infrastructure sous-jacente. Cette option convient aux utilisateurs qui ont besoin d'un contrôle granulaire et d'une personnalisation de leurs nœuds et qui sont prêts à investir du temps dans la gestion et la maintenance de leur infrastructure.

Concepts Kubernetes

Amazon Elastic Kubernetes Service (Amazon EKS) AWS est un service géré basé sur le projet open source. Kubernetes Bien que vous deviez savoir certaines choses sur la manière dont le service Amazon EKS s'intègre au AWS cloud (en particulier lorsque vous créez un cluster Amazon EKS pour la première fois), une fois qu'il est opérationnel, vous utilisez votre cluster Amazon EKS de la même manière que n'importe quel autre Kubernetes cluster. Pour commencer à gérer des Kubernetes clusters et à déployer des charges de travail, vous devez donc au moins avoir une connaissance de base des Kubernetes concepts.

Cette page divise Kubernetes les concepts en trois sections : PourquoiKubernetes, Clusters et Charges de travail. La première section décrit l'intérêt de l'exécution d'un Kubernetes service, en

particulier en tant que service géré tel qu'Amazon EKS. La section Charges de travail explique comment les Kubernetes applications sont créées, stockées, exécutées et gérées. La section Clusters présente les différents composants qui constituent les Kubernetes clusters et quelles sont vos responsabilités en matière de création et de maintenance de Kubernetes clusters.

Rubriques

Au fur et à mesure que vous parcourez ce contenu, des liens vous redirigeront vers des descriptions supplémentaires des Kubernetes concepts contenus dans Amazon EKS et dans la Kubernetes documentation, au cas où vous souhaiteriez approfondir l'un des sujets abordés ici. Pour en savoir plus sur la manière dont Amazon EKS implémente Kubernetes le plan de contrôle et les fonctionnalités de calcul, consultez l'architecture Amazon EKS.

Pourquoi choisir Kubernetes ?

Kubernetesa été conçu pour améliorer la disponibilité et l'évolutivité lors de l'exécution d'applications conteneurisées critiques de qualité pour la production. Plutôt que de simplement les exécuter Kubernetes sur une seule machine (bien que cela soit possible), vous Kubernetes atteignez ces objectifs en vous permettant d'exécuter des applications sur des ensembles d'ordinateurs qui peuvent s'étendre ou se contracter pour répondre à la demande. Kubernetesinclut des fonctionnalités qui vous permettent de :

- Déployer des applications sur plusieurs machines (à l'aide de conteneurs déployés dans des pods)
- Surveillez l'état des conteneurs et redémarrez les conteneurs défaillants
- Agrandir ou diminuer les conteneurs en fonction de la charge
- Mettre à jour les conteneurs avec les nouvelles versions
- Allouer des ressources entre les conteneurs
- Équilibrez le trafic entre les machines

L'Kubernetesautomatisation de ces types de tâches complexes permet aux développeurs d'applications de se concentrer sur la création et l'amélioration de leurs charges de travail, plutôt que de se soucier de l'infrastructure. Le développeur crée généralement des fichiers de configuration,

au format YAML, qui décrivent l'état souhaité de l'application. Cela peut inclure les conteneurs à exécuter, les limites de ressources, le nombre de répliques de Pod, l'allocation du processeur et de la mémoire, les règles d'affinité, etc.

Attributs de Kubernetes

Pour atteindre ses objectifs, Kubernetes possède les attributs suivants :

- Containerized - Kubernetes est un outil d'orchestration de conteneurs. Pour pouvoir l'utiliserKubernetes, vous devez d'abord avoir conteneurisé vos applications. Selon le type d'application, il peut s'agir d'un ensemble de microservices, de tâches par lots ou d'autres formes. Vos applications peuvent ensuite tirer parti d'un Kubernetes flux de travail qui englobe un vaste écosystème d'outils, dans lequel les conteneurs peuvent être stockés sous forme d'images dans un registre de conteneurs, déployés sur un Kubernetes cluster et exécutés sur un nœud disponible. Vous pouvez créer et tester des conteneurs individuels sur votre ordinateur local avec Docker ou sans autre environnement d'exécution de conteneur, avant de les déployer sur votre Kubernetes cluster.
- Évolutif : si la demande pour vos applications dépasse la capacité des instances en cours d'exécution de ces applications, Kubernetes vous pouvez augmenter votre capacité. Le cas échéant, Kubernetes peut déterminer si les applications ont besoin de plus de processeur ou de mémoire et réagir en augmentant automatiquement la capacité disponible ou en utilisant une plus grande partie de la capacité existante. Le dimensionnement peut être effectué au niveau du pod, s'il y a suffisamment de calcul disponible pour exécuter un plus grand nombre d'instances de l'application (scalage automatique du pod horizontal), ou au niveau du nœud, si davantage de nœuds doivent être créés pour gérer l'augmentation de capacité (Cluster Autoscaler ou Karpenter). Comme la capacité n'est plus nécessaire, ces services peuvent supprimer les pods inutiles et arrêter les nœuds inutiles.
- Disponible : si une application ou un nœud devient défaillant ou indisponible, Kubernetes vous pouvez déplacer les charges de travail en cours vers un autre nœud disponible. Vous pouvez résoudre le problème en supprimant simplement une instance en cours d'exécution d'une charge de travail ou d'un nœud qui exécute vos charges de travail. En fin de compte, les charges de travail peuvent être transférées sur d'autres sites si elles ne peuvent plus fonctionner là où elles se trouvent.
- Déclaratif : Kubernetes utilise la réconciliation active pour vérifier en permanence que l'état que vous déclarez pour votre cluster correspond à l'état réel. En appliquant Kubernetesdes objets à un cluster, généralement par le biais de fichiers de configuration au format YAML, vous pouvez, par exemple, demander de démarrer les charges de travail que vous souhaitez exécuter sur votre cluster. Vous pouvez ultérieurement modifier les configurations pour utiliser une version

ultérieure d'un conteneur ou allouer plus de mémoire, par exemple. Kubernetesfera ce qu'il faut pour établir l'état souhaité. Cela peut inclure l'augmentation ou la désactivation des nœuds, l'arrêt et le redémarrage des charges de travail ou le retrait de conteneurs mis à jour.

- Composable : étant donné qu'une application est généralement composée de plusieurs composants, vous souhaitez être en mesure de gérer ensemble un ensemble de ces composants (souvent représentés par plusieurs conteneurs). Bien que Docker Compose offre un moyen de le faire directement avecDocker, la commande Kubernetes Kompose peut vous aider à le faire avecKubernetes. Voir Translate a Docker Compose File to Kubernetes Resources pour un exemple de procédure à suivre.
- Extensible - Contrairement aux logiciels propriétaires, le Kubernetes projet open source est conçu pour vous permettre de l'étendre Kubernetes comme vous le souhaitez pour répondre à vos besoins. Les API et les fichiers de configuration sont ouverts aux modifications directes. Les tiers sont encouragés à créer leurs propres contrôleurs, afin d'étendre à la fois l'infrastructure et les fonctionnalités de l'utilisateur finalKubernetes. Les webhooks vous permettent de configurer des règles de cluster pour appliquer les politiques et vous adapter à l'évolution des conditions. Pour plus d'idées sur la manière d'étendre les Kubernetes clusters, consultez la section Extension Kubernetes.
- Portable - De nombreuses entreprises ont standardisé leurs opérations, Kubernetes car cela leur permet de gérer tous leurs besoins en matière d'applications de la même manière. Les développeurs peuvent utiliser les mêmes pipelines pour créer et stocker des applications conteneurisées. Ces applications peuvent ensuite être déployées sur des Kubernetes clusters exécutés sur site, dans le cloud, sur des point-of-sales terminaux de restaurants ou sur des appareils IoT dispersés sur les sites distants de l'entreprise. Sa nature open source permet aux utilisateurs de développer ces Kubernetes distributions spéciales, ainsi que les outils nécessaires pour les gérer.

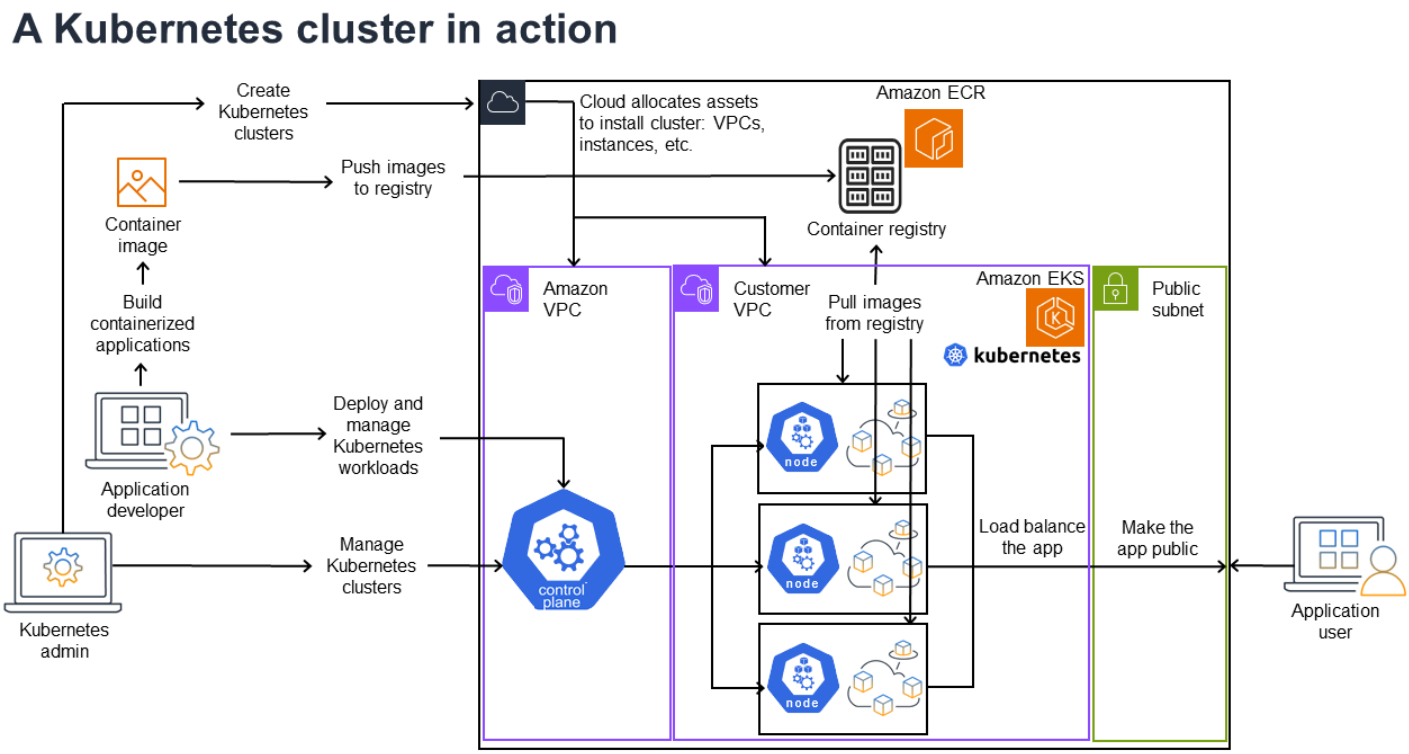
Gestion de Kubernetes

Kubernetesle code source est disponible gratuitement, donc avec votre propre équipement, vous pouvez l'installer et le gérer Kubernetes vous-même. Cependant, l'autogestion Kubernetes nécessite une expertise opérationnelle approfondie et sa maintenance demande du temps et des efforts. Pour ces raisons, la plupart des personnes qui déploient des charges de travail de production choisissent un fournisseur cloud (tel qu'Amazon EKS) ou un fournisseur sur site (tel qu'Amazon EKS Anywhere) avec sa propre Kubernetes distribution testée et le support d'Kubernetesexperts. Cela vous permet de vous décharger d'une grande partie des tâches lourdes indifférenciées nécessaires à la maintenance de vos clusters, notamment :

- Matériel : si vous ne disposez pas de matériel adapté à vos Kubernetes besoins, un fournisseur de cloud tel qu' AWS Amazon EKS peut vous faire économiser sur les coûts initiaux. Avec Amazon EKS, cela signifie que vous pouvez utiliser les meilleures ressources cloud proposées AWS, notamment les instances informatiques (Amazon Elastic Compute Cloud), votre propre environnement privé (Amazon VPC), la gestion centralisée des identités et des autorisations (IAM) et le stockage (Amazon EBS). AWS gère les ordinateurs, les réseaux, les centres de données et tous les autres composants physiques nécessaires au fonctionnementKubernetes. De même, vous n'avez pas à planifier votre centre de données de manière à gérer la capacité maximale les jours où la demande est la plus forte. Pour Amazon EKS Anywhere ou d'autres Kubernetes clusters sur site, vous êtes responsable de la gestion de l'infrastructure utilisée dans vos Kubernetes déploiements, mais vous pouvez toujours compter sur vous AWS pour vous aider à rester Kubernetes à jour.
- Gestion du plan de contrôle : Amazon EKS gère la sécurité et la disponibilité du plan de Kubernetes contrôle AWS hébergé, qui est chargé de planifier les conteneurs, de gérer la disponibilité des applications et d'autres tâches clés, afin que vous puissiez vous concentrer sur les charges de travail de vos applications. Si votre cluster tombe en panne, vous AWS devriez avoir les moyens de rétablir son état de fonctionnement. Pour Amazon EKS Anywhere, vous devez gérer vous-même le plan de contrôle.
- Mises à niveau testées : lorsque vous mettez à niveau vos clusters, vous pouvez compter sur Amazon EKS ou Amazon EKS Anywhere pour fournir des versions testées de leurs Kubernetes distributions.
- Modules complémentaires : il existe des centaines de projets conçus pour être étendus et Kubernetes compatibles. Vous pouvez les ajouter à l'infrastructure de votre cluster ou les utiliser pour faciliter l'exécution de vos charges de travail. Au lieu de créer et de gérer vous-même ces modules complémentaires, AWS nous proposons des modules complémentaires Amazon EKS que vous pouvez utiliser avec vos clusters. Amazon EKS Anywhere propose des packages sélectionnés qui incluent des versions de nombreux projets open source populaires. Vous n'avez donc pas à créer vous-même le logiciel ni à gérer les correctifs de sécurité critiques, les corrections de bogues ou les mises à niveau. De même, si les valeurs par défaut répondent à vos besoins, il est courant que très peu de configuration de ces modules complémentaires soient nécessaires. Voir Étendre les clusters pour plus de détails sur l'extension de votre cluster avec des modules complémentaires.

Kubernetesen action

Le schéma suivant montre les principales activités que vous devez effectuer en tant qu'Kubernetesadministrateur ou développeur d'applications pour créer et utiliser un Kubernetes cluster. Ce faisant, il illustre la manière dont Kubernetes les composants interagissent les uns avec les autres, en utilisant le AWS cloud comme exemple du fournisseur de cloud sous-jacent.



Un Kubernetes administrateur crée le Kubernetes cluster à l'aide d'un outil spécifique au type de fournisseur sur lequel le cluster sera construit. Cet exemple utilise le AWS cloud comme fournisseur, qui propose le Kubernetes service géré appelé Amazon EKS. Le service géré alloue automatiquement les ressources nécessaires à la création du cluster, notamment en créant deux nouveaux clouds privés virtuels (Amazon VPC) pour le cluster, en configurant le réseau, en mappant les Kubernetes autorisations nécessaires à la gestion des actifs dans le cloud, en veillant à ce que les services du plan de contrôle aient des emplacements où s'exécuter et en allouant zéro ou plusieurs instances Amazon EC2 en tant que nœuds pour exécuter les charges de travail. Kubernetes AWS gère lui-même un Amazon VPC pour le plan de contrôle, tandis que l'autre Amazon VPC contient les nœuds clients qui exécutent les charges de travail.

À l'avenir, de nombreuses tâches de l'Kubernetesadministrateur sont effectuées à l'aide d'Kubernetesoutils tels que kubectl. Cet outil envoie les demandes de services directement au plan

de contrôle du cluster. La manière dont les requêtes et les modifications sont apportées au cluster est alors très similaire à celle que vous utiliseriez pour n'importe quel Kubernetes cluster.

Un développeur d'applications souhaitant déployer des charges de travail sur ce cluster peut effectuer plusieurs tâches. Le développeur doit intégrer l'application dans une ou plusieurs images de conteneur, puis transférer ces images vers un registre de conteneurs accessible au Kubernetes cluster. AWS propose l'Amazon Elastic Container Registry (Amazon ECR) à cette fin.

Pour exécuter l'application, le développeur peut créer des fichiers de configuration au format YAML qui indiquent au cluster comment exécuter l'application, y compris les conteneurs à extraire du registre et comment les encapsuler dans des pods. Le plan de contrôle (planificateur) planifie les conteneurs sur un ou plusieurs nœuds et le moteur d'exécution des conteneurs sur chaque nœud extrait et exécute les conteneurs nécessaires. Le développeur peut également configurer un équilibreur de charge d'application pour équilibrer le trafic vers les conteneurs disponibles exécutés sur chaque nœud et exposer l'application afin qu'elle soit disponible sur un réseau public au monde extérieur. Une fois tout cela fait, une personne souhaitant utiliser l'application peut se connecter au point de terminaison de l'application pour y accéder.

La section suivante décrit en détail chacune de ces fonctionnalités, du point de vue des Kubernetes clusters et des charges de travail.

Clusters

Si votre travail consiste à démarrer et à gérer Kubernetes des clusters, vous devez savoir comment les Kubernetes clusters sont créés, améliorés, gérés et supprimés. Vous devez également savoir quels sont les composants qui constituent un cluster et ce que vous devez faire pour les maintenir à jour.

Les outils de gestion des clusters gèrent le chevauchement entre les Kubernetes services et le fournisseur de matériel sous-jacent. C'est pourquoi l'automatisation de ces tâches est généralement effectuée par le Kubernetes fournisseur (comme Amazon EKS ou Amazon EKS Anywhere) à l'aide d'outils spécifiques au fournisseur. Par exemple, pour démarrer un cluster Amazon EKS, vous pouvez l'utiliser`eksctl create cluster`, tandis que pour Amazon EKS Anywhere, vous pouvez utiliser`eksctl anywhere create cluster`. Notez que bien que ces commandes créent un Kubernetes cluster, elles sont spécifiques au fournisseur et ne font pas partie du Kubernetes projet lui-même.

Outils de création et de gestion de clusters

Le Kubernetes projet propose des outils permettant de créer un Kubernetes cluster manuellement. Ainsi, si vous souhaitez effectuer une installation Kubernetes sur une seule machine ou exécuter le plan de contrôle sur une machine et ajouter des nœuds manuellement, vous pouvez utiliser des outils CLI tels que kind, minikube ou kubeadm répertoriés sous Outils d'installation. Kubernetes Pour simplifier et automatiser le cycle de vie complet de la création et de la gestion des clusters, il est beaucoup plus facile d'utiliser des outils pris en charge par un Kubernetes fournisseur établi, tel qu'Amazon EKS ou Amazon EKS Anywhere.

Dans AWS le cloud, vous pouvez créer des clusters Amazon EKS à l'aide d'outils CLI, tels que eksctl, ou d'outils plus déclaratifs, tels que Terraform (voir Amazon EKS Blueprints for Terraform). Vous pouvez également créer un cluster à partir de la console AWS de gestion. Consultez les fonctionnalités d'Amazon EKS pour obtenir une liste de ce que vous pouvez obtenir avec Amazon EKS. KubernetesLes responsabilités qu'Amazon EKS assume pour vous incluent :

- Plan de contrôle géré : AWS garantit que le cluster Amazon EKS est disponible et évolutif, car il gère le plan de contrôle pour vous et le rend disponible dans toutes les zones de AWS disponibilité.
- Gestion des nœuds : au lieu d'ajouter manuellement des nœuds, vous pouvez demander à Amazon EKS de créer des nœuds automatiquement selon vos besoins, à l'aide de Managed Node Groups ou de Karpenter. Les groupes de nœuds gérés sont intégrés à Kubernetes Cluster Autoscaling. À l'aide des outils de gestion des nœuds, vous pouvez réaliser des économies, notamment grâce aux instances Spot, à la consolidation et à la disponibilité des nœuds, en utilisant les fonctionnalités de planification pour définir la manière dont les charges de travail sont déployées et les nœuds sélectionnés.
- Mise en réseau de clusters : à l'aide de CloudFormation modèles, `eksctl` configure la mise en réseau entre les composants du plan de contrôle et du plan de données (nœud) du Kubernetes cluster. Il met également en place des points de terminaison par lesquels les communications internes et externes peuvent avoir lieu. Consultez Démystifier le réseau de clusters pour les nœuds de travail Amazon EKS pour plus de détails. Les communications entre les pods dans Amazon EKS sont effectuées à l'aide d'Amazon EKS Pod Identities, qui permet aux Pods de tirer parti des méthodes AWS cloud de gestion des informations d'identification et des autorisations.
- Modules complémentaires : Amazon EKS vous évite d'avoir à créer et à ajouter des composants logiciels couramment utilisés pour prendre en charge les Kubernetes clusters. Par exemple, lorsque vous créez un cluster Amazon EKS à partir de la console de AWS gestion, celui-ci ajoute automatiquement le kube-proxy Amazon EKS, le plug-in Amazon VPC CNI pour et les modules complémentaires CoreDNS. Kubernetes Consultez les modules complémentaires Amazon EKS pour en savoir plus sur ces modules complémentaires, y compris la liste des modules disponibles.

Pour exécuter vos clusters sur vos propres ordinateurs et réseaux locaux, Amazon propose Amazon EKS Anywhere. Au lieu que le AWS Cloud soit le fournisseur, vous avez le choix d'exécuter Amazon EKS Anywhere sur les plateformes VMware vSphere, bare metal (fournisseur Tinkerbell) CloudStack, Snow ou Nutanix en utilisant votre propre équipement.

Amazon EKS Anywhere est basé sur le même logiciel Amazon EKS Distro que celui utilisé par Amazon EKS. Cependant, Amazon EKS Anywhere s'appuie sur différentes implémentations de l'interface CAPI (KubernetesCluster API) pour gérer le cycle de vie complet des machines d'un cluster Amazon EKS Anywhere (comme CAPV pour vSphere et CAPC pour). CloudStack Comme l'ensemble du cluster fonctionne sur votre équipement, vous assumez la responsabilité supplémentaire de gérer le plan de contrôle et de sauvegarder ses données (voir etcd plus loin dans ce document).

Composants du cluster

Kubernetesles composants du cluster sont divisés en deux zones principales : le plan de contrôle et les nœuds de travail. Les composants du plan de contrôle gèrent le cluster et fournissent un accès à ses API. Les nœuds de travail (parfois simplement appelés nœuds) fournissent les emplacements où les charges de travail réelles sont exécutées. Les composants du nœud sont des services qui s'exécutent sur chaque nœud pour communiquer avec le plan de contrôle et exécuter des conteneurs. L'ensemble des nœuds de travail de votre cluster est appelé plan de données.

Plan de contrôle

Le plan de contrôle est constitué d'un ensemble de services qui gèrent le cluster. Ces services peuvent tous être exécutés sur un seul ordinateur ou peuvent être répartis sur plusieurs ordinateurs. En interne, elles sont appelées instances du plan de contrôle (CPI). La manière dont les CPI sont exécutés dépend de la taille du cluster et des exigences en matière de haute disponibilité. À mesure que la demande augmente dans le cluster, un service de plan de contrôle peut évoluer pour fournir davantage d'instances de ce service, les demandes étant équilibrées entre les instances.

Les tâches effectuées par les composants du plan Kubernetes de contrôle sont les suivantes :

- Communication avec les composants du cluster (serveur d'API) : le serveur d'API (kube-apiserver) expose l'KubernetesAPI afin que les demandes puissent être adressées au cluster à la fois depuis l'intérieur et l'extérieur du cluster. En d'autres termes, les demandes d'ajout ou de modification des objets d'un cluster (pods, services, nœuds, etc.) peuvent provenir de commandes externes, telles que des demandes `kubectl` d'exécution d'un pod. De même, des demandes peuvent être

adressées par le serveur API aux composants du cluster, par exemple une requête au `kubelet` service concernant l'état d'un Pod.

- Stocker les données relatives au cluster (etcd key value store) - Le service etcd joue un rôle essentiel dans le suivi de l'état actuel du cluster. Si le service etcd devenait inaccessible, vous ne pourriez pas mettre à jour ou demander l'état du cluster, mais les charges de travail continueraient à s'exécuter pendant un certain temps. Pour cette raison, les clusters critiques disposent généralement de plusieurs instances équilibrées du service etcd exécutées simultanément et effectuent des sauvegardes périodiques de la banque de valeurs des clés etcd en cas de perte ou de corruption de données. N'oubliez pas que, dans Amazon EKS, tout cela est géré automatiquement par défaut. Amazon EKS Anywhere fournit des instructions pour la sauvegarde et la restauration etcd. Consultez le modèle de données etcd pour savoir comment etcd gère les données.
- Planifier des pods vers des nœuds (planificateur) - Les demandes de démarrage ou d'arrêt d'un pod Kubernetes sont dirigées vers le planificateur (Kuberneteskube-scheduler). Étant donné qu'un cluster peut comporter plusieurs nœuds capables d'exécuter le Pod, il appartient au planificateur de choisir sur quel nœud (ou quels nœuds, dans le cas de répliques) le Pod doit s'exécuter. Si la capacité disponible est insuffisante pour exécuter le Pod demandé sur un nœud existant, la demande échouera, sauf si vous avez pris d'autres dispositions. Ces dispositions pourraient inclure des services d'activation tels que les groupes de nœuds gérés ou Karpenter qui peuvent automatiquement démarrer de nouveaux nœuds pour gérer les charges de travail.
- Maintenir les composants dans l'état souhaité (Controller Manager) - Le Kubernetes Controller Manager s'exécute comme un processus daemon (kube-controller-manager) pour surveiller l'état du cluster et apporter des modifications au cluster afin de rétablir les états attendus. En particulier, plusieurs contrôleurs surveillent différents Kubernetes objets, notamment un statefulset-controller node-lifecycle-controller, un endpoint-controller, un cronjob-controller, etc.
- Gérer les ressources du cloud (Cloud Controller Manager) : les interactions entre le fournisseur de cloud Kubernetes et le fournisseur de cloud qui exécute les demandes relatives aux ressources du centre de données sous-jacent sont gérées par le Cloud Controller Manager (cloud-controller-manager). Les contrôleurs gérés par le Cloud Controller Manager peuvent inclure un contrôleur de route (pour configurer les itinéraires du réseau cloud), un contrôleur de service (pour utiliser les services d'équilibrage de charge du cloud) et un contrôleur de nœuds (pour utiliser des API cloud pour synchroniser les Kubernetes nœuds avec les nœuds cloud).

Nœuds de travail (plan de données)

Pour un Kubernetes cluster à nœud unique, les charges de travail s'exécutent sur la même machine que le plan de contrôle. Cependant, une configuration plus normale consiste à avoir un ou plusieurs systèmes informatiques distincts (nœuds) dédiés à l'exécution des Kubernetes charges de travail.

Lorsque vous créez un Kubernetes cluster pour la première fois, certains outils de création de cluster vous permettent de configurer un certain nombre de nœuds à ajouter au cluster (soit en identifiant les systèmes informatiques existants, soit en demandant au fournisseur d'en créer de nouveaux). Avant que des charges de travail ne soient ajoutées à ces systèmes, des services sont ajoutés à chaque nœud pour implémenter les fonctionnalités suivantes :

- Gérer chaque nœud (kubelet) : le serveur d'API communique avec le service kubelet exécuté sur chaque nœud pour s'assurer que le nœud est correctement enregistré et que les pods demandés par le planificateur sont en cours d'exécution. Le kubelet peut lire les manifestes des pods et configurer des volumes de stockage ou d'autres fonctionnalités nécessaires aux pods sur le système local. Il peut également vérifier l'état des conteneurs gérés localement.
- Exécuter des conteneurs sur un nœud (runtime de conteneur) : le Container Runtime de chaque nœud gère les conteneurs demandés pour chaque pod attribué au nœud. Cela signifie qu'il peut extraire les images du conteneur du registre approprié, exécuter le conteneur, l'arrêter et répondre aux requêtes concernant le conteneur. Le runtime du conteneur par défaut est containerd. À partir de la Kubernetes version 1.24, l'intégration spéciale de Docker (Dockershim) qui pouvait être utilisée comme environnement d'exécution du conteneur a été supprimée. Kubernetes Bien que vous puissiez toujours l'utiliser Docker pour tester et exécuter des conteneurs sur votre système local, pour l'utiliserDocker, Kubernetes vous devez désormais installer le Docker moteur sur chaque nœud pour l'utiliserKubernetes.
- Gérer le réseau entre les conteneurs (kube-proxy) - Pour pouvoir prendre en charge la communication entre les pods à l'aide des services, Kubernetes il fallait un moyen de configurer les réseaux de pods pour suivre les adresses IP et les ports associés à ces pods. Le service kube-proxy s'exécute sur chaque nœud pour permettre la communication entre les pods.

Étendre les clusters

Certains services peuvent être ajoutés pour Kubernetes prendre en charge le cluster, mais ils ne sont pas exécutés dans le plan de contrôle. Ces services s'exécutent souvent directement sur les nœuds de l'espace de noms du système Kube ou dans son propre espace de noms (comme c'est souvent le cas avec les fournisseurs de services tiers). Le service CoreDNS, qui fournit des services DNS au

cluster, en est un exemple courant. Reportez-vous à la section Découverte des services intégrés pour savoir comment voir quels services de cluster s'exécutent dans Kube-System sur votre cluster.

Il existe différents types de modules complémentaires que vous pouvez envisager d'ajouter à vos clusters. Pour préserver la santé de vos clusters, vous pouvez ajouter des fonctionnalités d'observabilité qui vous permettent d'effectuer des tâches telles que la journalisation, l'audit et les métriques. Grâce à ces informations, vous pouvez résoudre les problèmes qui surviennent, souvent par le biais des mêmes interfaces d'observabilité. Des exemples de ces types de services incluent Amazon GuardDuty, AWS Distro pour CloudWatchOpenTelemetry, le plugin Amazon VPC CNI pour et Grafana Kubernetes Monitoring. Kubernetes Pour le stockage, les modules complémentaires d'Amazon EKS incluent le pilote Amazon Elastic Block Store CSI (pour ajouter des périphériques de stockage par blocs), le pilote Amazon Elastic File System CSI (pour ajouter du stockage au système de fichiers) et plusieurs modules complémentaires de stockage tiers (tels que le pilote Amazon FSx pour NetApp ONTAP CSI).

Pour une liste plus complète des extensions Amazon EKS disponibles, consultez la section Extensions Amazon EKS.

Charges de travail

Kubernetesdéfinit une charge de travail comme « une application exécutée sur »Kubernetes. Cette application peut consister en un ensemble de microservices exécutés sous forme de conteneurs dans des pods, ou peut être exécutée sous forme de traitement par lots ou d'autres types d'applications. Le travail Kubernetes consiste à s'assurer que les demandes que vous faites pour la configuration ou le déploiement de ces objets sont exécutées. En tant que personne déployant des applications, vous devez apprendre comment les conteneurs sont créés, comment les pods sont définis et quelles méthodes vous pouvez utiliser pour les déployer.

Conteneurs

L'élément le plus élémentaire d'une charge de travail d'application que vous déployez et gérez Kubernetes est un pod. Un Pod représente un moyen de contenir les composants d'une application et de définir des spécifications décrivant les attributs du Pod. Comparez cela à quelque chose comme un package RPM ou Deb, qui regroupe des logiciels pour un système Linux, mais ne s'exécute pas lui-même en tant qu'entité.

Le Pod étant la plus petite unité déployable, il contient généralement un seul conteneur. Cependant, plusieurs conteneurs peuvent se trouver dans un Pod dans les cas où les conteneurs sont étroitement couplés. Par exemple, un conteneur de serveur Web peut être empaqueté dans un pod

avec un conteneur de type sidecar qui peut fournir des services de journalisation, de surveillance ou tout autre service étroitement lié au conteneur de serveur Web. Dans ce cas, le fait d'être dans le même Pod garantit que pour chaque instance en cours d'exécution du Pod, les deux conteneurs s'exécutent toujours sur le même nœud. De même, tous les conteneurs d'un Pod partagent le même environnement, les conteneurs d'un Pod s'exécutant comme s'ils se trouvaient sur le même hôte isolé. Cela a pour effet que les conteneurs partagent une adresse IP unique qui donne accès au Pod et que les conteneurs peuvent communiquer entre eux comme s'ils s'exécutaient sur leur propre hôte local.

Les spécifications du pod (PodSpec) définissent l'état souhaité du pod. Vous pouvez déployer un pod individuel ou plusieurs pods en utilisant les ressources de charge de travail pour gérer les modèles de pods. Les ressources de charge de travail incluent les déploiements (pour gérer plusieurs répliques de pods), StatefulSets(pour déployer des pods qui doivent être uniques, tels que les pods de base de données) et DaemonSets(où un pod doit fonctionner en continu sur chaque nœud). Plus d'informations à ce sujet plus tard.

Alors qu'un pod est la plus petite unité que vous déployez, un conteneur est la plus petite unité que vous créez et gérez.

Conteneurs de construction

Le Pod n'est en réalité qu'une structure autour d'un ou de plusieurs conteneurs, chaque conteneur contenant lui-même le système de fichiers, les exécutables, les fichiers de configuration, les bibliothèques et les autres composants nécessaires à l'exécution de l'application. Parce qu'une société appelée Docker Inc. a d'abord popularisé les conteneurs, certaines personnes appellent les conteneurs des Docker conteneurs. Cependant, l'Open Container Initiative a depuis défini les durées d'exécution des conteneurs, les images et les méthodes de distribution pour le secteur. Ajoutez à cela le fait que les conteneurs ont été créés à partir de nombreuses fonctionnalités Linux existantes, d'autres les appellent souvent conteneurs OCI, conteneurs Linux ou simplement conteneurs.

Lorsque vous créez un conteneur, vous commencez généralement par un Docker fichier (littéralement nommé ainsi). Dans ce Dockerfile, vous identifiez :

- Une image de base : une image de conteneur de base est un conteneur généralement créé à partir d'une version minimale du système de fichiers d'un système d'exploitation (tel que Red Hat Enterprise Linux ou Ubuntu) ou d'un système minimal amélioré pour fournir des logiciels permettant d'exécuter des types d'applications spécifiques (tels que des applications nodejs ou python).
- Logiciel d'application - Vous pouvez ajouter votre logiciel d'application à votre conteneur de la même manière que vous l'ajouteriez à un système Linux. Par exemple, dans votre Dockerfile, vous

pouvez exécuter `npm` et `yarn` installer une application Java ou `dnf` installer `yum` des packages RPM. En d'autres termes, à l'aide d'une commande RUN dans un Dockerfile, vous pouvez exécuter n'importe quelle commande disponible dans le système de fichiers de votre image de base pour installer un logiciel ou configurer un logiciel à l'intérieur de l'image de conteneur obtenue.

- Instructions - La référence Dockerfile décrit les instructions que vous pouvez ajouter à un Dockerfile lorsque vous le configurez. Il s'agit notamment des instructions utilisées pour créer le contenu du conteneur lui-même (`ADD`ou des `COPY` fichiers du système local), identifier les commandes à exécuter lorsque le conteneur est exécuté (`CMD`ou`ENTRYPOINT`) et connecter le conteneur au système sur lequel il s'exécute (en identifiant le `USER` conteneur sous lequel exécuter, un local `VOLUME` à monter ou les ports vers lesquels`EXPOSE`).

Bien que la `docker` commande et le service soient traditionnellement utilisés pour créer des conteneurs (`docker build`), d'autres outils sont disponibles pour créer des images de conteneurs, notamment podman et nerdctl. Voir Créer de meilleures images de conteneurs ou Construire avec Docker pour en savoir plus sur la création de conteneurs.

Conteneurs de stockage

Une fois que vous avez créé votre image de conteneur, vous pouvez la stocker dans un registre de distribution de conteneurs sur votre poste de travail ou dans un registre de conteneurs public. La gestion d'un registre de conteneurs privé sur votre poste de travail vous permet de stocker des images de conteneurs localement, les rendant ainsi facilement accessibles.

Pour stocker des images de conteneurs de manière plus publique, vous pouvez les transférer vers un registre de conteneurs public. Les registres publics de conteneurs fournissent un emplacement central pour le stockage et la distribution des images des conteneurs. Les registres de conteneurs publics incluent notamment le registre Amazon Elastic Container Registry, le registre Red Hat Quay et le registre DockerHub.

Lorsque vous exécutez des charges de travail conteneurisées sur Amazon Elastic Kubernetes Service (Amazon EKS), nous vous recommandons de récupérer des copies des images officielles stockées dans Amazon Docker Elastic Container Registry. AWS Amazon ECR stocke ces images depuis 2021. Vous pouvez rechercher des images de conteneurs populaires dans la galerie publique Amazon ECR, et plus particulièrement pour les images du Docker Hub, vous pouvez effectuer une recherche dans la galerie Amazon ECR. Docker

Conteneurs de course

Les conteneurs étant conçus dans un format standard, ils peuvent être exécutés sur n'importe quelle machine capable d'exécuter un environnement d'exécution de conteneur (telle queDocker) et dont le contenu correspond à l'architecture de la machine locale (telle que `x86_64` ou`arm`). Pour tester un conteneur ou simplement l'exécuter sur votre bureau local, vous pouvez utiliser `podman run` les commandes `docker run` or pour démarrer un conteneur sur l'hôte local. CependantKubernetes, chaque nœud de travail dispose d'un environnement d'exécution de conteneur déployé et il appartient Kubernetes à un nœud de demander à ce qu'il exécute un conteneur.

Une fois qu'un conteneur a été assigné pour s'exécuter sur un nœud, le nœud vérifie si la version demandée de l'image du conteneur existe déjà sur le nœud. Si ce n'est pas le cas, Kubernetes indique au moteur d'exécution du conteneur d'extraire ce conteneur du registre de conteneurs approprié, puis de l'exécuter localement. N'oubliez pas qu'une image de conteneur fait référence au package logiciel qui est déplacé entre votre ordinateur portable, le registre des conteneurs et Kubernetes les nœuds. Un conteneur fait référence à une instance en cours d'exécution de cette image.

Capsules

Une fois que vos conteneurs sont prêts, l'utilisation des pods inclut la configuration, le déploiement et la mise à disposition des pods.

Configuration des pods

Lorsque vous définissez un Pod, vous lui attribuez un ensemble d'attributs. Ces attributs doivent inclure au moins le nom du pod et l'image du conteneur à exécuter. Cependant, vous souhaitez également configurer de nombreux autres éléments avec les définitions de votre pod (consultez la PodSecpage pour plus de détails sur ce qui peut être inclus dans un pod). Il s'agit des licences suivantes :

- Stockage : lorsqu'un conteneur en cours d'exécution est arrêté et supprimé, le stockage des données dans ce conteneur disparaît, sauf si vous configurez un stockage plus permanent. Kubernetesprend en charge de nombreux types de stockage différents et les regroupe sous l'égide de Volumes. Les types de stockage incluent CephFS, NFS, iSCSI et autres. Vous pouvez même utiliser un périphérique de blocage local à partir de l'ordinateur local. Avec l'un de ces types de stockage disponible sur votre cluster, vous pouvez monter le volume de stockage sur un point de montage sélectionné dans le système de fichiers de votre conteneur. Un volume persistant est un volume qui continue d'exister après la suppression du pod, tandis qu'un volume éphémère

est supprimé lorsque le pod est supprimé. Si votre administrateur de cluster a créé une solution différente StorageClassespour votre cluster, vous pouvez choisir les attributs du stockage que vous utilisez, par exemple si le volume est supprimé ou récupéré après utilisation, s'il sera étendu si davantage d'espace est nécessaire, et même s'il répond à certaines exigences de performance.

- Secrets - En mettant les secrets à la disposition des conteneurs dans les spécifications du Pod, vous pouvez fournir les autorisations dont ces conteneurs ont besoin pour accéder aux systèmes de fichiers, aux bases de données ou à d'autres actifs protégés. Les clés, les mots de passe et les jetons font partie des éléments qui peuvent être stockés en tant que secrets. L'utilisation de secrets vous évite d'avoir à stocker ces informations dans des images de conteneurs, mais vous n'avez qu'à les mettre à la disposition des conteneurs en cours d'exécution. Similaire à Secrets are ConfigMaps. A ConfigMap a tendance à contenir des informations moins critiques, telles que les paires clé-valeur pour configurer un service.
- Ressources de conteneur - Les objets permettant de configurer davantage les conteneurs peuvent prendre la forme d'une configuration de ressources. Pour chaque conteneur, vous pouvez demander la quantité de mémoire et de processeur qu'il peut utiliser, ainsi que fixer des limites à la quantité totale de ces ressources que le conteneur peut utiliser. Voir Gestion des ressources pour les pods et les conteneurs pour des exemples.
- Interruptions - Les pods peuvent être perturbés involontairement (un nœud tombe en panne) ou volontairement (une mise à niveau est souhaitée). En configurant un budget d'interruption du Pod, vous pouvez exercer un certain contrôle sur la disponibilité de votre application en cas de perturbations. Consultez la section Spécification d'un budget d'interruption pour votre application pour des exemples.
- Espaces de noms : Kubernetes propose différentes méthodes pour isoler les Kubernetes composants et les charges de travail les uns des autres. L'exécution de tous les pods d'une application particulière dans le même espace de noms est une méthode courante pour sécuriser et gérer ces pods ensemble. Vous pouvez créer vos propres espaces de noms à utiliser ou choisir de ne pas indiquer d'espace de noms (ce qui entraîne l'utilisation de l'espace Kubernetes de `default` noms). Kubernetesles composants du plan de contrôle s'exécutent généralement dans l'espace de noms du système Kube.

La configuration qui vient d'être décrite est généralement regroupée dans un fichier YAML à appliquer au Kubernetes cluster. Pour les Kubernetes clusters personnels, vous pouvez simplement stocker ces fichiers YAML sur votre système local. Cependant, avec des clusters et des charges de travail plus critiques, GitOpsc'est un moyen populaire d'automatiser le stockage et les mises à jour des charges de travail et des ressources Kubernetes d'infrastructure.

Les objets utilisés pour rassembler et déployer les informations du Pod sont définis par l'une des méthodes de déploiement suivantes.

Déploiement de pods

La méthode que vous choisirez pour déployer des pods dépend du type d'application que vous prévoyez d'exécuter avec ces pods. Voici certains de vos choix :

- Applications sans état : une application sans état n'enregistre pas les données de session d'un client. Une autre session n'a donc pas besoin de se référer à ce qui s'est passé lors d'une session précédente. Cela permet de remplacer plus facilement les pods par de nouveaux s'ils deviennent insalubres ou de les déplacer sans enregistrer leur état. Si vous exécutez une application sans état (telle qu'un serveur Web), vous pouvez utiliser un déploiement pour déployer des pods et ReplicaSets. A ReplicaSet définit le nombre d'instances d'un Pod que vous souhaitez exécuter simultanément. Bien que vous puissiez exécuter un pod ReplicaSet directement, il est courant d'exécuter des répliques directement dans un déploiement, afin de définir le nombre de répliques d'un pod à exécuter à la fois.
- Applications dynamiques - Une application dynamique est une application dans laquelle l'identité du Pod et l'ordre dans lequel les Pods sont lancés sont importants. Ces applications ont besoin d'un stockage persistant stable qui doit être déployé et mis à l'échelle de manière cohérente. Pour déployer une application dynamique dansKubernetes, vous pouvez utiliser StatefulSets. Une base de données est un exemple d'application généralement exécutée en tant StatefulSet que. Dans un StatefulSet, vous pouvez définir les répliques, le Pod et ses conteneurs, les volumes de stockage à monter et les emplacements dans le conteneur où les données sont stockées. Voir Exécuter une application dynamique répliquée pour un exemple de déploiement d'une base de données en tant que. ReplicaSet
- Applications par nœud : il arrive que vous souhaitiez exécuter une application sur chaque nœud de votre Kubernetes cluster. Par exemple, votre centre de données peut exiger que chaque ordinateur exécute une application de surveillance ou un service d'accès à distance spécifique. En Kubernetes effet, vous pouvez utiliser a DaemonSetpour vous assurer que l'application sélectionnée s'exécute sur tous les nœuds de votre cluster.
- Les applications s'exécutent jusqu'à la fin : vous souhaitez exécuter certaines applications pour effectuer une tâche particulière. Cela peut inclure un système qui publie des rapports d'état mensuels ou nettoie les anciennes données. Un objet Job peut être utilisé pour configurer une application afin qu'elle démarre et s'exécute, puis qu'elle se ferme une fois la tâche terminée. Un CronJobobjet vous permet de configurer une application pour qu'elle s'exécute à une heure, une minute, un jour du mois, un mois ou un jour de la semaine spécifiques, en utilisant une structure définie par le format crontab Linux.

Rendre les applications accessibles depuis le réseau

Les applications étant souvent déployées sous la forme d'un ensemble de microservices qui se déplaçaient vers différents endroits, Kubernetes il fallait trouver un moyen pour que ces microservices puissent se trouver les uns les autres. En outre, pour que d'autres puissent accéder à une application en dehors du Kubernetes cluster, Kubernetes il fallait trouver un moyen d'exposer cette application sur des adresses et des ports externes. Ces fonctionnalités liées au réseau sont réalisées avec les objets Service et Ingress, respectivement :

- Services - Comme un Pod peut se déplacer vers différents nœuds et adresses, un autre Pod ayant besoin de communiquer avec le premier Pod peut avoir du mal à le trouver. Pour résoudre ce problème, vous Kubernetes permet de représenter une application en tant que service. Avec un service, vous pouvez identifier un pod ou un ensemble de pods avec un nom particulier, puis indiquer quel port expose le service de cette application depuis le pod et quels ports une autre application pourrait utiliser pour contacter ce service. Un autre Pod au sein d'un cluster peut simplement demander un service par son nom et Kubernetes redirigera cette demande vers le port approprié pour une instance du Pod exécutant ce service.
- Entrée - L'entrée est ce qui permet de mettre les applications représentées par Kubernetes des services à la disposition des clients extérieurs au cluster. Les fonctionnalités de base d'Ingress incluent un équilibreur de charge (géré par Ingress), le contrôleur Ingress et des règles de routage des demandes du contrôleur vers le Service. Il existe plusieurs contrôleurs d'entrée parmi lesquels vous pouvez choisir. Kubernetes

Étapes suivantes

Comprendre Kubernetes les concepts de base et leur lien avec Amazon EKS vous aidera à parcourir à la fois la documentation Amazon EKS et la Kubernetesdocumentation afin de trouver les informations dont vous avez besoin pour gérer les clusters Amazon EKS et déployer des charges de travail sur ces clusters. Pour commencer à utiliser Amazon EKS, choisissez l'une des options suivantes :

- Création d'un cluster simple
- Création d'un cluster plus complexe
- Déployer un exemple d'application
- Découvrez les moyens de gérer votre cluster

Options de déploiement

Vous pouvez déployer Amazon EKS avec l'une des options suivantes :

Amazon EKS dans le cloud

Vous pouvez exécuter Kubernetes dans le cloud AWS sans avoir besoin d'installer, d'exploiter et de gérer votre propre plan de contrôle ou nœuds Kubernetes. C'est cette option qui est abordée dans ce guide.

Amazon EKS sur Outposts

AWS Outposts active les Services AWS natifs, l'infrastructure et les modèles d'exploitation dans vos installations sur site. Avec Amazon EKS sur Outposts, vous pouvez choisir d'exécuter des clusters étendus ou locaux. Avec des clusters étendus, le plan de contrôle Kubernetes s'exécute dans une Région AWS et les nœuds s'exécutent sur des Outposts. Avec des clusters locaux, l'ensemble du cluster Kubernetes s'exécute localement sur Outposts, y compris le plan de contrôle et les nœuds Kubernetes. Pour de plus amples informations, veuillez consulter Amazon EKS sur AWS Outposts.

Amazon EKS Anywhere

Amazon EKS Anywhere est une option de déploiement pour Amazon EKS qui vous permet de créer et d'exploiter facilement des clusters Kubernetes sur site. Amazon EKS et Amazon EKS Anywhere sont tous deux basés sur Amazon EKS Distro. Pour en savoir plus sur Amazon EKS Anywhere et ses différences avec Amazon EKS, consultez Présentation et Comparer Amazon EKS Anywhere à Amazon EKS dans la documentation Amazon EKS Anywhere. Pour obtenir des réponses à certaines questions courantes, voirFAQ sur Amazon EKS Anywhere.

Amazon EKS Distro

Amazon EKS Distro est une distribution des mêmes logiciels et dépendances Kubernetes open source déployés par Amazon EKS dans le cloud. Amazon EKS Distro suit le même cycle de publication de version Kubernetes qu'Amazon EKS et est fourni en tant que projet open source. Pour en savoir plus, consultez Amazon EKS Distro. Vous pouvez également afficher et télécharger le code source de Amazon EKS Distro sur GitHub.

Lorsque vous choisissez les options de déploiement à utiliser pour votre cluster Kubernetes, tenez compte des éléments suivants :

Fonctionnalité	Amazon EKS	Amazon EKS sur Outposts	Amazon EKS Anywhere	Amazon EKS Distro
Matériel	AWS : fourni	AWS : fourni	Fourni par vous	Fourni par vous
Emplacement de déploiement	Cloud AWS	Votre centre de données	Votre centre de données	Votre centre de données
Emplacement du plan de contrôle Kubernetes	Cloud AWS	AWS Cloud ou votre centre de données	Votre centre de données	Votre centre de données
Emplacement du plan de données Kubernetes	Cloud AWS	Votre centre de données	Votre centre de données	Votre centre de données
Support	AWS Support	AWS Support	AWS Support	Support de la communauté OSS

Configuration de l'utilisation d'Amazon EKS

Les ressources AWS sont généralement soumises à des restrictions qui limitent l'accès à l'entité AWS qui les a créées. Par conséquent, il est essentiel d'établir une configuration utilisateur appropriée dans l'AWS Command Line Interface dès le début. En outre, vous devez équiper votre ordinateur local d'outils essentiels pour une gestion efficace par ligne de commande de votre cluster Amazon EKS. Cette rubrique vous aidera à préparer la gestion par ligne de commande de votre cluster.

Étape 1 : Configuration de l'AWS CLI

L'AWS CLI est un outil de ligne de commande pour travailler avec les services AWS, y compris Amazon EKS. Il sert également à authentifier les utilisateurs ou les rôles IAM afin d'accéder au cluster Amazon EKS et à d'autres ressources AWS depuis votre ordinateur local. Pour mettre en service des ressources dans AWS depuis la ligne de commande, vous devez obtenir un ID de clé d'accès AWS et une clé secrète à utiliser dans la ligne de commande. Vous devez ensuite configurer ces informations d'identification dans l'AWS CLI. Si vous n'avez pas installé l'AWS CLI, consultez Installation ou mise à jour de la dernière version de l'AWS CLI dans le Guide de l'utilisateur de l'AWS Command Line Interface.

Pour créer une clé d'accès

1. Connectez-vous au AWS Management Console.
2. En haut à droite, choisissez votre nom d'utilisateur AWS pour ouvrir le menu de navigation. Par exemple, sélectionnez **`webadmin`**. Choisissez ensuite Mes identifiants de sécurité.
3. Sous Clés d'accès, choisissez Créer une clé d'accès.
4. Choisissez Interface de ligne de commande (CLI), puis Suivant.
5. Choisissez Create access key (Créer une clé d'accès).
6. Choisissez Télécharger le fichier .csv.

Pour configurer l'AWS CLI

Après avoir téléchargé l'AWS CLI, procédez comme suit pour la configurer. Pour plus d'informations, voir la rubrique Configuration de la AWS CLI du Guide de l'utilisateur de la AWS Command Line Interface.

1. Dans une fenêtre de terminal, entrez la commande suivante :

   ```
   aws configure
   ```

   Vous pouvez éventuellement configurer un profil nommé, tel que **`--profile cluster-admin`**. Si vous configurez un profil nommé dans l'AWS CLI, vous devez toujours transmettre cet indicateur dans les commandes suivantes.

2. Saisissez vos informations d'identification AWS. Par exemple :

   ```
   AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   Default region name [None]: region-code
   Default output format [None]: json
   ```

Pour obtenir un jeton de sécurité

Si nécessaire, exécutez la commande suivante pour obtenir un nouveau jeton de sécurité pour l'AWS CLI. Pour plus d'informations, consultez la section `get-session-token` dans la référence des commandes AWS CLI.

Par défaut, le jeton est valide pendant 15 minutes. Pour modifier le délai d'expiration de session par défaut, transmettez l'indicateur **`--duration-seconds`**. Par exemple :

```
aws sts get-session-token --duration-seconds 3600
```

Cette commande renvoie les informations d'identification de sécurité temporaires pour une session AWS CLI. Le résultat de la réponse devrait être le suivant :

```
{
    "Credentials": {
        "AccessKeyId": "ASIA5FTRU3LOEXAMPLE",
        "SecretAccessKey": "JnKgvwfqUD9mNsPoi9IbxAYEXAMPLE",
        "SessionToken": "VERYLONGSESSIONTOKENSTRING",
        "Expiration": "2023-02-17T03:14:24+00:00"
    }
}
```

Pour vérifier l'identité de l'utilisateur

Si nécessaire, exécutez la commande suivante pour vérifier les informations d'identification AWS de l'utilisateur IAM (par exemple *ClusterAdmin*) pour la session du terminal.

```
aws sts get-caller-identity
```

Cette commande renvoie l'Amazon Resource Name (ARN) de l'entité IAM configurée pour l'AWS CLI. Vous devriez voir l'exemple de réponse suivant :

```
{
    "UserId": "AKIAIOSFODNN7EXAMPLE",
    "Account": "01234567890",
    "Arn": "arn:aws:iam::01234567890:user/ClusterAdmin"
}
```

Étape 2 : Installation des outils Kubernetes

Pour communiquer avec un cluster Kubernetes, vous aurez besoin d'un outil permettant d'interagir avec l'API Kubernetes. En outre, vous avez besoin de quelques autres outils, tels qu'un outil pour gérer les environnements Kubernetes sur votre ordinateur local.

Pour créer des ressources AWS

- Ressources du cluster Amazon EKS : si vous n'êtes pas familier avec AWS, nous vous recommandons d'installer `eksctl`. `eksctl` est un utilitaire d'infrastructure en tant que code (IaC) qui utilise AWS CloudFormation pour créer facilement votre cluster Amazon EKS. Il crée également des ressources Kubernetes supplémentaires, telles que des comptes de service. Pour les instructions d'installation de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.
- Ressources AWS : si vous avez l'habitude d'automatiser le provisionnement et le déploiement de votre infrastructure AWS, nous vous recommandons d'installer Terraform. Terraform est un outil d'infrastructure en tant que code (IaC) open source développé par HashiCorp. Il vous permet de définir et de provisionner l'infrastructure à l'aide d'un langage de configuration de haut niveau tel que HashiCorp Configuration Language (HCL) ou JSON. Pour les instructions d'installation de Terraform, consultez la rubrique Installation de Terraform dans la documentation Terraform.

Pour installer **kubectl**

`kubectl` est un outil de ligne de commande open source utilisé pour communiquer avec le serveur d'API Kubernetes de votre cluster Amazon EKS. Si vous ne l'avez pas encore installé sur votre ordinateur local, choisissez l'une des options suivantes :

- Versions AWS : pour installer une version `kubectl` compatible avec Amazon EKS, consultez Installation ou mise à jour de `kubectl`.
- Versions communautaires : pour installer la dernière version communautaire de `kubectl`, consultez la page Outils d'installation de la documentation Kubernetes.

Pour configurer un environnement de développement

- Outil de déploiement local : si vous n'êtes pas familier avec Kubernetes, envisagez d'installer un outil de déploiement local tel que `minikube` ou `kind`. Ces outils vous permettent de gérer un cluster Amazon EKS sur votre ordinateur local.
- Gestionnaire de packages : Helm est un gestionnaire de packages populaire pour Kubernetes qui simplifie l'installation et la gestion de packages complexes. Grâce à Helm, il est plus facile d'installer et de gérer des packages tels que le contrôleur d'équilibreur de charge AWS sur votre cluster Amazon EKS.

Étapes suivantes

- Démarrer avec Amazon EKS

Installation ou mise à jour de **kubectl**

`Kubectl` est un outil de ligne de commande que vous utilisez pour communiquer avec le serveur d'API Kubernetes. Le fichier binaire `kubectl` est disponible dans de nombreux gestionnaires de packages de systèmes d'exploitation. L'utilisation d'un gestionnaire de packages pour votre installation est souvent plus facile que celle d'un processus d'installation et de téléchargement manuel.

Cette rubrique vous aide à télécharger et installer ou mettre à jour le fichier binaire `kubectl` sur votre appareil. Le fichier binaire est identique aux versions communautaires en amont. Le binaire n'est pas propre à Amazon EKS ou AWS.

> **Note**
>
> Vous devez utiliser une version `kubectl` qui se situe à une différence de version mineure près du plan de contrôle de votre cluster Amazon EKS. Par exemple, un client `kubectl` `1.29` doit utiliser des clusters Kubernetes, `1.28`, `1.29` et `1.30`.

Pour installer ou mettre à jour **`kubectl`**

1. Déterminez si vous avez déjà `kubectl` installé sur votre appareil.

   ```
   kubectl version --client
   ```

   Si `kubectl` est installé dans le chemin d'accès de votre appareil, l'exemple de sortie comprend des informations similaires à celles qui suivent. Si vous souhaitez mettre à jour la version que vous avez actuellement installée avec une version plus récente, passez à l'étape suivante en vous assurant d'installer la nouvelle version au même emplacement que votre version actuelle.

   ```
   Client Version: v1.30.X-eks-1234567
   ```

   Si vous n'obtenez aucune sortie, vous n'avez pas `kubectl` installé, ou il n'est pas installé dans un emplacement qui se trouve sur le chemin d'accès de votre appareil.
2. Installation ou mise à jour de `kubectl` sur les systèmes d'exploitation `macOS`, Linux, et Windows.

   macOS

   Pour installer ou mettre à jour **`kubectl`** sur **`macOS`**

   1. Téléchargez le fichier binaire pour la version Kubernetes de votre cluster depuis Amazon S3.
      - Kubernetes `1.30`

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/darwin/amd64/kubectl
        ```

      - Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/
darwin/amd64/kubectl
```

- Kubernetes 1.28

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/
darwin/amd64/kubectl
```

- Kubernetes 1.27

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/
bin/darwin/amd64/kubectl
```

- Kubernetes 1.26

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/
bin/darwin/amd64/kubectl
```

- Kubernetes 1.25

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/
bin/darwin/amd64/kubectl
```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/
bin/darwin/amd64/kubectl
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/
bin/darwin/amd64/kubectl
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/
bin/darwin/amd64/kubectl
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/
bin/darwin/amd64/kubectl
```

2. (Facultatif) Vérifiez le fichier binaire téléchargé avec le total de contrôle SHA-256 de votre fichier binaire.

   a. Télécharger le total de contrôle SHA-256 pour la version Kubernetes de votre cluster.

   - Kubernetes 1.30

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/
     bin/darwin/amd64/kubectl.sha256
     ```

   - Kubernetes 1.29

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/
     bin/darwin/amd64/kubectl.sha256
     ```

   - Kubernetes 1.28

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/
     bin/darwin/amd64/kubectl.sha256
     ```

   - Kubernetes 1.27

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/
     bin/darwin/amd64/kubectl.sha256
     ```

   - Kubernetes 1.26

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/
     bin/darwin/amd64/kubectl.sha256
     ```

   - Kubernetes 1.25

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/
     bin/darwin/amd64/kubectl.sha256
     ```

   - Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

b. Vérifiez le total de contrôle `SHA-256` du fichier binaire téléchargé.

```
openssl sha1 -sha256 kubectl
```

c. Assurez-vous que le total de contrôle généré dans la sortie correspond à celui du fichier `kubectl.sha256` téléchargé.

3. Appliquez les autorisations d'exécution au fichier binaire.

```
chmod +x ./kubectl
```

4. Copiez le fichier binaire vers un dossier de votre `PATH`. Si vous avez déjà installé une version de `kubectl`, nous vous recommandons de créer un fichier `$HOME/bin/kubectl` et de vérifier que le fichier `$HOME/bin` est en premier dans votre `$PATH`.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Facultatif) Ajoutez le chemin `$HOME/bin` à votre fichier d'initialisation du shell de façon à ce qu'il soit configuré lorsque vous ouvrez un shell.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bash_profile
```

Linux (amd64)

Pour installer ou mettre à jour **`kubectl`** sur Linux (**`amd64`**)

1. Téléchargez le fichier binaire `kubectl` pour la version Kubernetes de votre cluster depuis Amazon S3.
   - Kubernetes `1.30`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.30`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.29`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.28`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.27`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.26`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/amd64/kubectl
```

2. (Facultatif) Vérifiez le fichier binaire téléchargé avec le total de contrôle `SHA-256` de votre fichier binaire.

   a. Téléchargez le total de contrôle `SHA-256` pour la version Kubernetes de votre cluster depuis Amazon S3 à l'aide de la commande correspondant à la plateforme matérielle de votre appareil. Le premier lien pour chaque version est pour `amd64` et le second lien est pour `arm64`.

   - Kubernetes `1.30`

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl.sha256
   ```

   - Kubernetes `1.30`

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl.sha256
   ```

   - Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/
bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/amd64/kubectl.sha256
```

b. Vérifiez le total de contrôle SHA-256 du fichier binaire que vous avez téléchargé à l'aide de l'une des commandes suivantes.

- ```
  sha256sum -c kubectl.sha256
  ```

  Lorsque vous utilisez cette commande, assurez-vous que la sortie suivante s'affiche :

  ```
  kubectl: OK
  ```

- ```
  openssl sha1 -sha256 kubectl
  ```

  Lorsque vous utilisez cette commande, assurez-vous que le total de contrôle généré dans la sortie correspond à celui du fichier `kubectl.sha256` téléchargé.

3. Appliquez les autorisations d'exécution au fichier binaire.

```
chmod +x ./kubectl
```

4. Copiez le fichier binaire vers un dossier de votre PATH. Si vous avez déjà installé une version de `kubectl`, nous vous recommandons de créer un fichier `$HOME/bin/kubectl` et de vérifier que le fichier `$HOME/bin` est en premier dans votre `$PATH`.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Facultatif) Ajoutez le chemin `$HOME/bin` à votre fichier d'initialisation du shell de façon à ce qu'il soit configuré lorsque vous ouvrez un shell.

> **Note**
>
> Cette étape suppose que vous utilisiez le shell Bash ; si vous utilisez un autre shell, modifiez la commande pour utiliser votre propre fichier d'initialisation shell.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
```

Linux (arm64)

Pour installer ou mettre à jour **`kubectl`** sur Linux (**`arm64`**)

1. Téléchargez le fichier binaire `kubectl` pour la version Kubernetes de votre cluster depuis Amazon S3.
   - Kubernetes `1.30`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.30`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.29`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.28`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.27`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.26`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/arm64/kubectl
```

2. (Facultatif) Vérifiez le fichier binaire téléchargé avec le total de contrôle `SHA-256` de votre fichier binaire.

   a. Téléchargez le total de contrôle `SHA-256` pour la version Kubernetes de votre cluster depuis Amazon S3 à l'aide de la commande correspondant à la plateforme matérielle de votre appareil. Le premier lien pour chaque version est pour `amd64` et le second lien est pour `arm64`.

   - Kubernetes `1.30`

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl.sha256
   ```

   - Kubernetes `1.30`

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl.sha256
   ```

   - Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/
bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/arm64/kubectl.sha256
```

b. Vérifiez le total de contrôle SHA-256 du fichier binaire que vous avez téléchargé à l'aide de l'une des commandes suivantes.

- ```
  sha256sum -c kubectl.sha256
  ```

  Lorsque vous utilisez cette commande, assurez-vous que la sortie suivante s'affiche :

  ```
  kubectl: OK
  ```

- ```
  openssl sha1 -sha256 kubectl
  ```

  Lorsque vous utilisez cette commande, assurez-vous que le total de contrôle généré dans la sortie correspond à celui du fichier `kubectl.sha256` téléchargé.

3. Appliquez les autorisations d'exécution au fichier binaire.

```
chmod +x ./kubectl
```

4. Copiez le fichier binaire vers un dossier de votre PATH. Si vous avez déjà installé une version de `kubectl`, nous vous recommandons de créer un fichier `$HOME/bin/kubectl` et de vérifier que le fichier `$HOME/bin` est en premier dans votre $PATH.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Facultatif) Ajoutez le chemin `$HOME/bin` à votre fichier d'initialisation du shell de façon à ce qu'il soit configuré lorsque vous ouvrez un shell.

> **Note**
>
> Cette étape suppose que vous utilisiez le shell Bash ; si vous utilisez un autre shell, modifiez la commande pour utiliser votre propre fichier d'initialisation shell.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
```

Windows

Pour installer ou mettre à jour **`kubectl`** sur Windows

1. Ouvrez un terminal PowerShell.
2. Téléchargez le fichier binaire `kubectl` pour la version Kubernetes de votre cluster depuis Amazon S3.
   - Kubernetes `1.30`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.29`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.28`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.27`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.26`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.25`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.24`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.24.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.23`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.22`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.21`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.21.14/2024-04-19/bin/windows/amd64/kubectl.exe
```

3. (Facultatif) Vérifiez le fichier binaire téléchargé avec le total de contrôle `SHA-256` de votre fichier binaire.

   a. Téléchargez le total de contrôle `SHA-256` pour la version Kubernetes de votre cluster pour Windows.

   - Kubernetes `1.30`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.30.0/2024-05-12/bin/windows/amd64/kubectl.exe.sha256
```

   - Kubernetes `1.29`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.29.3/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

   - Kubernetes `1.28`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.28.8/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

   - Kubernetes `1.27`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.27.12/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.26`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.26.15/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.25`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.25.16/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.24`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.24.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.23`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.22`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.21`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.21.14/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

b. Vérifiez le total de contrôle `SHA-256` du fichier binaire téléchargé.

```
Get-FileHash kubectl.exe
```

c. Assurez-vous que le total de contrôle généré dans la sortie correspond à celui du fichier `kubectl.sha256` téléchargé. La PowerShell sortie doit être une chaîne de caractères équivalente en majuscules.

4. Copiez le fichier binaire vers un dossier de votre PATH. Si vous disposez déjà d'un répertoire dans votre PATH que vous utilisez pour les utilitaires de ligne de commande, copiez le fichier binaire dans ce répertoire. Sinon, effectuez les étapes suivantes.
   a. Créez un répertoire pour vos fichiers binaires de ligne de commande, par exemple `C:\bin`.
   b. Copiez le fichier binaire `kubectl.exe` vers votre nouveau répertoire.
   c. Modifiez la variable d'environnement PATH de votre utilisateur ou système pour ajouter le nouveau répertoire à votre variable PATH.
   d. Fermez votre terminal PowerShell et ouvrez-en un nouveau pour sélectionner la nouvelle variable PATH.
3. Après avoir installé `kubectl`, vous pouvez vérifier sa version.

```
kubectl version --client
```

Lors de la première installation de `kubectl`, il n'est pas encore configuré pour communiquer avec un serveur. Nous aborderons cette configuration au besoin dans d'autres procédures. Si vous devez mettre à jour la configuration pour communiquer avec un cluster en particulier, vous pouvez exécuter la commande suivante. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

Démarrer avec Amazon EKS

Assurez-vous que votre configuration vous permet d'utiliser Amazon EKS avant de consulter les guides de démarrage. Pour plus d'informations, consultez Configuration de l'utilisation d'Amazon EKS.

Il existe deux guides de démarrage pour créer un nouveau cluster Kubernetes avec les nœuds dans Amazon EKS :

- Démarrage avec Amazon EKS : `eksctl` : ce guide de démarrage vous permet d'installer toutes les ressources nécessaires pour commencer à utiliser Amazon EKS avec `eksctl`, un utilitaire de ligne de commande simple pour créer et gérer les clusters Kubernetes sur Amazon EKS. À la fin de ce didacticiel, vous disposerez d'un cluster Amazon EKS en cours d'exécution sur lequel vous pouvez déployer des applications. Il s'agit de la solution la plus simple et la plus rapide pour commencer à utiliser Amazon EKS.
- Commencer à utiliser Amazon EKS — AWS Management Console et AWS CLI— Ce guide de démarrage vous aide à créer toutes les ressources nécessaires pour démarrer avec Amazon EKS à l'aide du AWS Management Console et AWS CLI. À la fin de ce didacticiel, vous disposerez d'un cluster Amazon EKS en cours d'exécution sur lequel vous pouvez déployer des applications. Dans ce guide, vous créez manuellement chaque ressource requise pour un cluster Amazon EKS. Les procédures vous donnent une visibilité sur la façon dont chaque ressource est créée et la manière dont elles interagissent les unes avec les autres.

Nous proposons également les références suivantes :

- Pour consulter une sélection de didacticiels pratiques, consultez Navigating Amazon EKS on AWS Community.
- Pour des exemples de code, consultez la section Exemples de code pour Amazon EKS à l'aide de AWS kits SDK.

Démarrage avec Amazon EKS : **`eksctl`**

Ce guide vous permet de créer toutes les ressources nécessaires pour commencer à utiliser Amazon Elastic Kubernetes Service (Amazon EKS) avec `eksctl`, un utilitaire de ligne de commande simple pour créer et gérer les clusters Kubernetes sur Amazon EKS. À la fin de ce didacticiel, vous

disposerez d'un cluster Amazon EKS en cours d'exécution sur lequel vous pouvez déployer des applications.

Les procédures de ce guide créent automatiquement plusieurs ressources que vous devez créer manuellement lorsque vous créez votre cluster à l'aide de l'outil AWS Management Console. Si vous préférez créer manuellement la plupart des ressources pour mieux comprendre comment elles interagissent les unes avec les autres, utilisez le AWS Management Console pour créer votre cluster et effectuer des calculs. Pour plus d'informations, consultez Commencer à utiliser Amazon EKS — AWS Management Console et AWS CLI.

Prérequis

Avant de démarrer ce didacticiel, vous devez installer et configurer les outils et les ressources suivants dont vous avez besoin pour créer et gérer un cluster Amazon EKS.

- **`kubectl`** : outil de ligne de commande pour travailler avec des clusters Kubernetes. Pour plus d'informations, consultez Installation ou mise à jour de `kubectl`.
- **`eksctl`** : outil de ligne de commande pour travailler avec des clusters EKS qui automatise plusieurs tâches individuelles. Pour plus d'informations, veuillez consulter Installation dans la documentation `eksctl`.
- Autorisations IAM requises : le principal de sécurité IAM que vous utilisez doit disposer des autorisations nécessaires pour utiliser les rôles IAM Amazon EKS, les rôles liés à un service, AWS CloudFormation un VPC et les ressources associées. Pour plus d'informations, consultez Actions, ressources et clés de condition pour Amazon Elastic Container Service for Kubernetes et Utilisation des rôles liés à un service dans le guide de l'utilisateur IAM. Vous devez effectuer toutes les étapes de ce guide avec le même utilisateur. Exécutez la commande suivante pour vérifier l'utilisateur actuel :

```
aws sts get-caller-identity
```

Étape 1 : créer votre cluster Amazon EKS et votre nœud

> **Important**
>
> Pour démarrer le plus simplement et le plus rapidement possible, cette rubrique inclut les étapes pour créer un cluster et des nœuds avec des paramètres par défaut. Avant de créer un cluster et des nœuds pour une utilisation en production, nous vous recommandons de

vous familiariser avec tous les paramètres et de déployer un cluster et des nœuds avec les paramètres qui répondent à vos besoins. Pour plus d'informations, consultez Création d'un cluster Amazon EKS et Nœuds Amazon EKS. Certains paramètres ne peuvent être activés que lors de la création de votre cluster et de vos nœuds.

Vous pouvez créer un cluster à l'aide de l'un des types de nœuds suivants. Pour en savoir plus sur chaque type, consultez Nœuds Amazon EKS. Une fois votre cluster déployé, vous pouvez ajouter d'autres types de nœuds.

- Fargate : Linux : sélectionnez ce type de nœud si vous voulez exécuter des applications Linux sur AWS Fargate. Fargate est un moteur de calcul sans serveur qui vous permet de déployer des Pods Kubernetes sans gérer les instances Amazon EC2.
- Nœuds gérés : Linux : sélectionnez ce type de nœud si vous souhaitez exécuter des applications Amazon Linux sur des instances Amazon EC2. Bien que cela ne soit pas inclus dans ce guide, vous pouvez également ajouter des nœuds autogérés par Windows et Bottlerocket à votre cluster.

Créez votre cluster Amazon EKS à l'aide de la commande suivante. Vous pouvez remplacer *my-cluster* par votre propre valeur. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. *region-code*Remplacez-le par tout Région AWS ce qui est pris en charge par Amazon EKS. Pour en obtenir la liste Régions AWS, consultez la section Points de terminaison et quotas Amazon EKS dans le guide de référence AWS général.

Fargate – Linux

```
eksctl create cluster --name my-cluster --region region-code --fargate
```

Managed nodes – Linux

```
eksctl create cluster --name my-cluster --region region-code
```

La création du cluster prend plusieurs minutes. Pendant la création, vous verrez plusieurs lignes de sortie. La dernière ligne de sortie est similaire à celle de l'exemple suivant.

```
[...]
```

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

`eksctl` a créé un fichier `kubectl config` dans `~/.kube` ou a ajouté la configuration du nouveau cluster dans un fichier `config` dans `~/.kube` sur votre ordinateur.

Une fois la création du cluster terminée, consultez la AWS CloudFormation pile nommée `eksctl-my-cluster-cluster` dans la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation pour voir toutes les ressources créées.

Étape 2 : afficher les ressources Kubernetes

1. Affichez vos nœuds de cluster.

   ```
   kubectl get nodes -o wide
   ```

   L'exemple qui suit illustre un résultat.

   Fargate – Linux

   ```
   NAME                                                  STATUS   ROLES    AGE
      VERSION              INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-
   VERSION                    CONTAINER-RUNTIME
   fargate-ip-192-0-2-0.region-code.compute.internal   Ready    <none>
     8m3s    v1.2.3-eks-1234567   192.0.2.0     <none>        Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   fargate-ip-192-0-2-1.region-code.compute.internal   Ready    <none>
     7m30s   v1.2.3-eks-1234567   192-0-2-1     <none>        Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ```

   Managed nodes – Linux

   ```
   NAME                                          STATUS   ROLES    AGE    VERSION
             INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-VERSION
          CONTAINER-RUNTIME
   ip-192-0-2-0.region-code.compute.internal   Ready    <none>   6m7s
      v1.2.3-eks-1234567   192.0.2.0     192.0.2.2     Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ip-192-0-2-1.region-code.compute.internal   Ready    <none>   6m4s
      v1.2.3-eks-1234567   192.0.2.1     192.0.2.3     Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ```

Pour plus d'informations sur les éléments que vous voyez en sortie, consultez Afficher les ressources Kubernetes.

2. Affichez les applications en cours d'exécution sur votre cluster.

```
kubectl get pods -A -o wide
```

L'exemple qui suit illustre un résultat.

Fargate – Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE   IP
          NODE                                                NOMINATED NODE
 READINESS GATES
kube-system   coredns-1234567890-abcde   1/1     Running   0          18m
  192.0.2.0   fargate-ip-192-0-2-0.region-code.compute.internal   <none>
   <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          18m
  192.0.2.1   fargate-ip-192-0-2-1.region-code.compute.internal   <none>
   <none>
```

Managed nodes – Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE     IP
          NODE                                        NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-12345             1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
kube-system   aws-node-67890             1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-abcde   1/1     Running   0          14m
   192.0.2.3   ip-192-0-2-3.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          14m
   192.0.2.4   ip-192-0-2-4.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-12345           1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
```

```
kube-system   kube-proxy-67890           1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
```

Pour plus d'informations sur les éléments que vous voyez en sortie, consultez Afficher les ressources Kubernetes.

Étape 3 : supprimer votre cluster et vos nœuds

Une fois que vous avez fini d'utiliser le cluster et les nœuds que vous avez créés pour ce didacticiel, vous devez effectuer un nettoyage en supprimant le cluster et les nœuds avec la commande suivante. Si vous souhaitez exécuter d'autres opérations avec ce cluster avant le nettoyage, consultez Étapes suivantes.

```
eksctl delete cluster --name my-cluster --region region-code
```

Étapes suivantes

Les rubriques suivantes de la documentation vous aideront à étendre les fonctionnalités de votre cluster.

- Déployez un exemple d'application sur votre cluster.
- Le principal IAM qui a créé le cluster est le seul principal à pouvoir effectuer des appels au serveur d'API Kubernetes à l'aide de `kubectl` ou de la AWS Management Console. Si vous souhaitez que d'autres principaux IAM aient accès à votre cluster, vous devez les ajouter. Pour plus d'informations, consultez Autoriser l'accès aux Kubernetes API et Autorisations nécessaires.
- Avant de déployer un cluster pour une utilisation en production, nous vous recommandons de vous familiariser avec tous les paramètres des clusters et des nœuds. Certains paramètres, tels que l'activation de l'accès SSH aux nœuds Amazon EC2, doivent être effectués lors de la création du cluster.
- Pour renforcer la sécurité de votre cluster, configurez le plugin Amazon VPC Container Networking Interface pour utiliser les rôles IAM pour les comptes de service.

Commencer à utiliser Amazon EKS — AWS Management Console et AWS CLI

Ce guide vous aide à créer toutes les ressources nécessaires pour démarrer avec Amazon Elastic Kubernetes Service (Amazon EKS) à l'aide du et du. AWS Management Console AWS CLI Dans ce guide, vous créez manuellement chaque ressource. À la fin de ce didacticiel, vous disposerez d'un cluster Amazon EKS en cours d'exécution sur lequel vous pouvez déployer des applications.

Les procédures de ce guide vous donnent une visibilité complète sur la façon dont chaque ressource est créée et dont les ressources interagissent les unes avec les autres. Si vous préférez créer automatiquement la plupart des ressources pour vous, utilisez la CLI `eksctl` pour créer votre cluster et vos nœuds. Pour plus d'informations, consultez Démarrage avec Amazon EKS : `eksctl`.

Prérequis

Avant de démarrer ce didacticiel, vous devez installer et configurer les outils et les ressources suivants dont vous avez besoin pour créer et gérer un cluster Amazon EKS.

- AWS CLI— Un outil de ligne de commande pour travailler avec AWS des services, notamment Amazon EKS. Pour plus d'informations, consultez Installation, mise à jour et désinstallation d' AWS CLI dans le Guide de l'utilisateur AWS Command Line Interface . Après l'avoir installé AWS CLI, nous vous recommandons de le configurer également. Pour plus d'informations, consultez Configuration rapide avec `aws configure` dans le Guide de l'utilisateur AWS Command Line Interface .
- **`kubectl`** : outil de ligne de commande pour travailler avec des clusters Kubernetes. Pour plus d'informations, consultez Installation ou mise à jour de `kubectl`.
- Autorisations IAM requises : le principal de sécurité IAM que vous utilisez doit être autorisé à utiliser les rôles IAM Amazon EKS, les rôles liés à un service, AWS CloudFormation un VPC et les ressources associées. Pour plus d'informations, consultez Actions, ressources et clés de condition pour Amazon Elastic Kubernetes Service et Utilisation des rôles liés à un service dans le guide de l'utilisateur IAM. Vous devez effectuer toutes les étapes de ce guide avec le même utilisateur. Exécutez la commande suivante pour vérifier l'utilisateur actuel :

  ```
  aws sts get-caller-identity
  ```

- Nous vous recommandons de terminer les étapes de cette rubrique dans un shell Bash. Si vous n'utilisez pas de shell Bash, certaines commandes de script telles que les caractères

de continuation de ligne et la façon dont les variables sont définies et utilisées nécessitent un ajustement pour votre shell. En outre, les règles de votre shell en matière de guillemets peuvent être différentes. Pour plus d'informations, consultez la section Utilisation de guillemets avec des chaînes AWS CLI dans le Guide de AWS Command Line Interface l'utilisateur.

Étape 1 : créer votre cluster Amazon EKS

> **Important**
>
> Pour démarrer le plus simplement et le plus rapidement possible, cette rubrique inclut les étapes pour créer un cluster avec des paramètres par défaut. Avant de créer un cluster pour une utilisation en production, nous vous recommandons de vous familiariser avec tous les paramètres et de déployer un cluster avec les paramètres qui répondent à vos besoins. Pour plus d'informations, consultez Création d'un cluster Amazon EKS. Certains paramètres ne peuvent être activés que lors de la création de votre cluster.

Pour créer votre cluster

1. Créez un Amazon VPC avec des sous-réseaux publics et privés qui répondent aux exigences Amazon EKS. Remplacez *region-code* par n'importe quelle Région AWS prise en charge par Amazon EKS. Pour en obtenir la liste Régions AWS, consultez la section Points de terminaison et quotas Amazon EKS dans le guide de référence AWS général.Vous pouvez remplacer *my-eks-vpc-stack* par n'importe quel nom que vous choisissez.

   ```
   aws cloudformation create-stack \
     --region region-code \
     --stack-name my-eks-vpc-stack \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
   ```

   > **Tip**
   >
   > Pour obtenir une liste de toutes les ressources créées par la commande précédente, ouvrez la console AWS CloudFormation à l'adresse https://console.aws.amazon.com/cloudformation. Choisissez la pile *my-eks-vpc-stack*, puis choisissez l'onglet Ressources.

2. Créez un rôle IAM de cluster et associez-y la politique gérée Amazon EKS IAM requise. Kubernetesles clusters gérés par Amazon EKS appellent d'autres AWS services en votre nom pour gérer les ressources que vous utilisez avec le service.

   a. Copiez le contenu suivant dans un fichier nommé *eks-cluster-role-trust-policy.json*.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "eks.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

   b. Créez le rôle.

   ```
   aws iam create-role \
     --role-name myAmazonEKSClusterRole \
     --assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
   ```

   c. Attachez la politique IAM gérée par Amazon EKS au rôle.

   ```
   aws iam attach-role-policy \
     --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
     --role-name myAmazonEKSClusterRole
   ```

3. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

   Assurez-vous que l'image Région AWS affichée dans le coin supérieur droit de votre console est Région AWS celle dans laquelle vous souhaitez créer votre cluster. Si ce n'est pas le cas, choisissez le menu déroulant à côté du Région AWS nom et choisissez celui Région AWS que vous souhaitez utiliser.

4. Sélectionnez Add cluster (Ajouter un cluster), puis Create (Créer). Si cette option ne s'affiche pas, sélectionnez d'abord Clusters dans le panneau de navigation gauche.

5. Sur la page Configure cluster (Configurer le cluster), procédez de la façon suivante :

   a. Saisissez un Name (Nom) pour votre cluster, tel que **`my-cluster`**. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.

   b. Pour Cluster Service Role, choisissez *`ClusterRoleMyAmazonEks`*.

   c. Laissez les autres paramètres à leurs valeurs par défaut et choisissez Next (Suivant).

6. Sur la page Specify networking (Spécifier les réseaux), procédez comme suit :

   a. Sélectionnez l'ID du VPC que vous avez créé à l'étape précédente dans la liste déroulante des VPC. Cela ressemble à *`vpc-00x0000x000x0x000`* | *`my-eks-vpc-stack-VPC`*.

   b. Laissez les autres paramètres à leurs valeurs par défaut et choisissez Next (Suivant).

7. Sur la page Configurer l'observabilité, cliquez sur Suivant.

8. Sur la page Sélectionner des modules complémentaires, choisissez Suivant.

   Pour plus d'informations sur les modules complémentaires, consultez Modules complémentaires Amazon EKS.

9. Sur la page Configurer les paramètres des modules complémentaires, choisissez Suivant.

10. Sur la page Review and create (Vérifier et créer), choisissez Create (Créer).

    À droite du nom du cluster, le cluster est en Creating (En cours de création) pendant plusieurs minutes jusqu'à la fin du processus d'approvisionnement du cluster. Ne passez à l'étape suivante que lorsque le cluster est Actif.

    > **Note**
    >
    > Il est possible que vous receviez un message d'erreur indiquant que l'une des zones de disponibilité de votre demande ne dispose pas d'une capacité suffisante pour créer un cluster Amazon EKS. Si cela se produit, la sortie de l'erreur contient les zones de disponibilité qui peuvent prendre en charge un nouveau cluster. Essayez à nouveau de créer votre cluster avec au moins deux sous-réseaux situés dans les zones de disponibilité prises en charge pour votre compte. Pour plus d'informations, consultez Capacité insuffisante.

Étape 2 : configurer votre ordinateur de façon à ce qu'il communique avec votre cluster

Dans cette section, vous créez un fichier `kubeconfig` pour votre cluster. Les paramètres de ce fichier activent l'option `kubectl` CLI pour communiquer avec votre cluster.

Pour configurer votre ordinateur pour qu'il communique avec votre cluster

1. Créez ou mettez à jour un fichier `kubeconfig` pour votre cluster. Remplacez *`region-code`* par la Région AWS dans laquelle vous souhaitez créer votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster.

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Par défaut, le paramètre `config` est créé dans `~/.kube` ou la configuration du nouveau cluster est ajoutée à un `config` fichier dans `~/.kube`.
2. Testez votre configuration.

   ```
   kubectl get svc
   ```

   > **Note**
   >
   > Si vous recevez d'autres erreurs concernant les types d'autorisations ou de ressources, consultez Accès non autorisé ou refusé (`kubectl`) dans la rubrique relative à la résolution des problèmes.

   L'exemple qui suit illustre un résultat.

   ```
   NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
   svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
   ```

Étape 3 : créer des nœuds

> **Important**
>
> Pour démarrer le plus simplement et le plus rapidement possible, cette rubrique inclut les étapes pour créer des nœuds avec des paramètres par défaut. Avant de créer des nœuds pour une utilisation en production, nous vous recommandons de vous familiariser avec tous les paramètres et de déployer des nœuds avec les paramètres qui répondent à vos besoins. Pour plus d'informations, consultez Nœuds Amazon EKS. Certains paramètres ne peuvent être activés que lors de la création de vos nœuds.

Vous pouvez créer un cluster à l'aide de l'un des types de nœuds suivants. Pour en savoir plus sur chaque type, consultez Nœuds Amazon EKS. Une fois votre cluster déployé, vous pouvez ajouter d'autres types de nœuds.

- Fargate : Linux : choisissez ce type de nœud si vous voulez exécuter des applications Linux sur AWS Fargate. Fargate est un moteur de calcul sans serveur qui vous permet de déployer des Pods Kubernetes sans gérer les instances Amazon EC2.
- Nœuds gérés : Linux : choisissez ce type de nœud si vous souhaitez exécuter des applications Amazon Linux sur des instances Amazon EC2. Bien que cela ne soit pas inclus dans ce guide, vous pouvez également ajouter des nœuds autogérés par Windows et Bottlerocket à votre cluster.

Fargate – Linux

Créez un profil Fargate. Lorsque les Pods Kubernetes sont déployés avec des critères qui correspondent aux critères définis dans le profil, les Pods sont déployés sur Fargate.

Pour créer un profil Fargate

1. Créez un rôle IAM et attachez-y la politique gérée Amazon EKS IAM requise. Lorsque votre cluster est créé Pods sur une infrastructure Fargate, les composants exécutés sur l'infrastructure Fargate doivent appeler les API en votre nom. AWS Cela leur permet d'effectuer des actions telles que l'extraction d'images de conteneurs depuis Amazon ECR ou le routage des journaux vers d'autres AWS services. Pour ce faire, le rôle d'exécution de Pod Amazon EKS fournit les autorisations IAM.

a. Copiez le contenu suivant dans un fichier nommé *pod-execution-role-trust-policy*.json. *region-code*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Si vous souhaitez utiliser le même rôle dans l'ensemble Régions AWS de votre compte, remplacez *region-code* par*. Remplacez *111122223333* par l'ID de votre compte et *my-cluster* par le nom de votre cluster. Si vous souhaitez utiliser le même rôle pour tous les clusters de votre compte, remplacez *my-cluster* par *.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Créez un rôle IAM d'exécution de Pod.

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-
policy.json"
```

c. Attachez la politique IAM gérée par Amazon EKS au rôle.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/
AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

2. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
3. Sur la page Clusters, choisissez le cluster *my-cluster*.
4. Sur la page ***my-cluster***, procédez comme suit :
   a. Choisissez l'onglet Calcul.
   b. Sous Fargate profiles (Profils Fargate), sélectionnez Add Fargate Profile (Ajouter un profil Fargate).
5. Sur la page Configure Fargate profile (Configurer le profil Fargate), procédez comme suit :
   a. Pour Nom, saisissez un nom unique pour votre profil Fargate, tel que ***my-profile***.
   b. Pour le rôle d'exécution du Pod, choisissez l'FargatePodExecutionRoleAmazoneks que vous avez créé à l'étape précédente.
   c. Sélectionnez le menu déroulant Sous-réseaux et désélectionnez tout sous-réseau dont le nom contient `Public`. Seuls les sous-réseaux privés sont pris en charge pour les Pods qui s'exécutent sur Fargate.
   d. Choisissez Suivant.
6. Sur la page Configurer la sélection de Pod, procédez comme suit :
   a. Pour Espace de noms, saisissez **`default`**.
   b. Choisissez Suivant.
7. Sur la page Vérifier et créer, vérifiez les informations de votre profil Fargate et choisissez Créer.
8. Après quelques minutes, le Status (Statut) de la section Fargate Profile configuration (Configuration du profil) Fargate passera de Creating (En cours de création) à Active (Actif). Ne passez à l'étape suivante que lorsque le cluster est Active (Actif).
9. Si vous prévoyez de déployer tous les Pods sur Fargate (et aucun sur les nœuds Amazon EC2), procédez comme suit pour créer un autre profil Fargate et exécuter le résolveur de noms par défaut (CoreDNS) sur Fargate.

> **Note**
>
> Si vous ne procédez pas ainsi, vous n'aurez aucun nœud à ce stade.

a. Sur la page Fargate Profile (Profil Fargate), choisissez *my-profile*.

b. Sous Fargate profiles (Profils Fargate), choisissez Add Fargate Profile (Ajouter un profil Fargate).

c. Pour Name (Nom), saisissez ***CoreDNS***.

d. Pour le rôle d'exécution du Pod, choisissez l'FargatePodExecutionRoleAmazoneks que vous avez créé à l'étape précédente.

e. Sélectionnez le menu déroulant Sous-réseaux et désélectionnez tout sous-réseau dont le nom contient `Public`. Seuls les sous-réseaux privés sont pris en charge pour les Pods exécutés sur Fargate.

f. Choisissez Suivant.

g. Pour Espace de noms, saisissez **`kube-system`**.

h. Choisissez Match labels (Faire correspondre les étiquettes), puis choisissez Add label (Ajouter une étiquette).

i. Saisissez **`k8s-app`** pour Key (Clé), et **`kube-dns`** pour la valeur. Cette étape est nécessaire pour que le résolveur de noms par défaut (CoreDNS) puisse être déployé sur Fargate.

j. Choisissez Suivant.

k. Sur la page Vérifier et créer, vérifiez les informations de votre profil Fargate et choisissez Créer.

l. Exécutez la commande suivante pour supprimer l'annotation `eks.amazonaws.com/compute-type : ec2` par défaut des Pods CoreDNS.

```
kubectl patch deployment coredns \
    -n kube-system \
    --type json \
    -p='[{"op": "remove", "path": "/spec/template/metadata/annotations/
eks.amazonaws.com~1compute-type"}]'
```

> **Note**
>
> Le système crée et déploie deux nœuds basés sur l'étiquette de profil Fargate que vous avez ajoutée. Vous ne verrez rien de répertorié dans Node Groups (Groupes

de nœuds) car ils ne sont pas applicables aux nœuds Fargate, mais vous verrez les nouveaux nœuds répertoriés dans l'onglet Overview (Présentation).

Managed nodes – Linux

Créez un groupe de nœuds géré, en spécifiant les sous-réseaux et le rôle IAM de nœud que vous avez créés au cours des étapes précédentes.

Pour créer votre groupe de nœuds gérés par Linux Amazon EC2

1. Créez un rôle IAM de nœud et attachez-y la politique gérée par Amazon EKS IAM requise. Le `kubelet` démon du nœud Amazon EKS passe des appels aux AWS API en votre nom. Les nœuds reçoivent l'autorisation pour ces appels d'API via un profil d'instance IAM et les politiques associées.

   a. Copiez le contenu suivant dans un fichier nommé *`node-role-trust-policy.json`*.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "ec2.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

   b. Créez le rôle IAM du nœud.

   ```
   aws iam create-role \
     --role-name myAmazonEKSNodeRole \
     --assume-role-policy-document file://"node-role-trust-policy.json"
   ```

   c. Attachez les politiques IAM gérées requises au rôle.

   ```
   aws iam attach-role-policy \
     --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
     --role-name myAmazonEKSNodeRole
   ```

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name myAmazonEKSNodeRole
```

2. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
3. Choisissez le nom du cluster que vous avez créé dans Étape 1 : créer votre cluster Amazon EKS, comme *my-cluster*.
4. Sur la page ***my-cluster***, procédez comme suit :
   a. Choisissez l'onglet Compute (Calcul).
   b. Choisissez Add Node Group (Ajouter un groupe de nœuds).
5. Sur la page Configure Node Group (Configurer le groupe de nœuds), procédez comme suit :
   a. Pour Name (Nom), saisissez un nom unique pour votre groupe de nœuds gérés, tel que ***my-nodegroup***. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.
   b. Pour le nom du rôle Node IAM, choisissez le NodeRole rôle *MyAmazonEKS* que vous avez créé à l'étape précédente. Nous recommandons que chaque groupe de nœuds utilise son propre rôle IAM unique.
   c. Choisissez Next (Suivant).
6. Sur la page Set compute and scaling configuration (Définir la configuration de calcul et de mise à l'échelle), acceptez les valeurs par défaut et choisissez Next (Suivant).
7. Sur la page Specify networking (Spécifier les réseaux), acceptez les valeurs par défaut et choisissez Next (Suivant).
8. Sur la page Review and create (Vérifier et créer), vérifiez la configuration de votre groupe de nœuds gérés et choisissez Create (Créer).
9. Après quelques minutes, le Status (Statut) dans la Node Group configuration (Configuration du groupe de nœuds) passera de Creating (En cours de création) à Active (Actif). Ne passez à l'étape suivante que lorsque le cluster est Active (Actif).

Étape 4 : afficher les ressources

Vous pouvez afficher vos nœuds et vos charges de travail Kubernetes.

Pour afficher vos nœuds et vos charges de travail

1. Dans le panneau de navigation de gauche, choisissez Clusters. Dans la liste Clusters, sélectionnez le nom du cluster que vous avez créé, tel que *my-cluster*.
2. Sur la page ***my-cluster***, procédez comme suit :
    a. Onglet Compute (Calcul) – La liste Nodes (Nœuds) qui ont été déployés pour le cluster s'affiche. Vous pouvez choisir le nom d'un nœud pour voir plus d'informations à son sujet.
    b. Onglet Ressources : toutes les ressources Kubernetes qui sont déployées par défaut sur un cluster Amazon EKS s'affichent. Sélectionnez un type de ressources dans la console pour en savoir plus.

Étape 5 : Suppression des ressources

Une fois que vous avez terminé avec le cluster et les nœuds que vous avez créés pour ce didacticiel, vous devez supprimer les ressources que vous avez créées. Si vous souhaitez exécuter d'autres opérations avec ce cluster de supprimer les ressources, consultez Étapes suivantes.

Pour supprimer les ressources que vous avez créées dans ce guide

1. Supprimez les groupes de nœuds ou les profils Fargate que vous avez créés.
    a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
    b. Dans le panneau de navigation de gauche, choisissez Clusters. Dans la liste des clusters, choisissez *my-cluster*.
    c. Choisissez l'onglet Calcul.
    d. Si vous avez créé un groupe de nœudsélectionnez le groupe de nœuds *my-nodegroup*, puis cliquez sur Supprimer. Entrez ***my-nodegroup***, puis choisissez Delete (Supprimer).
    e. Pour chaque profil Fargate que vous avez créé, choisissez-le, puis choisissez Delete (Supprimer). Saisissez le nom du profil, puis choisissez Delete (Supprimer).

> **Note**
>
> Lorsque vous supprimez un deuxième profil Fargate, vous devrez peut-être attendre que la suppression du premier profil soit terminée.

f. Ne continuez pas tant que le groupe de nœuds ou les profils Fargate supprimés.

2. Supprimez le cluster.

   a. Dans le panneau de navigation de gauche, choisissez Clusters. Dans la liste des clusters, choisissez *my-cluster*.

   b. Choisissez Supprimer le cluster.

   c. Saisissez ***my-cluster*** (supprimer), puis sélectionnez Delete (Supprimer). Ne continuez pas jusqu'à ce que le cluster soit supprimé.

3. Supprimez la AWS CloudFormation pile VPC que vous avez créée.

   a. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

   b. Choisissez la pile *my-eks-vpc-stack*, puis choisissez Supprimer.

   c. Dans la boîte de dialogue de confirmation Delete (Supprimer)***my-eks-vpc-stack***, choisissez Delete stack (Supprimer la pile).

4. Supprimez le rôle IAM que vous avez créé.

   a. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

   b. Dans le panneau de navigation de gauche, choisissez Rôles.

   c. ***Sélectionnez chaque rôle que vous avez créé dans la liste (MyAmazoneksClusterRole, ainsi que Amazoneks ou MyAmazoneks). FargatePod ExecutionRole NodeRole*** Choisissez Delete (Supprimer), saisissez le texte de confirmation demandé, puis choisissez Delete (Supprimer).

Étapes suivantes

Les rubriques suivantes de la documentation vous aideront à étendre les fonctionnalités de votre cluster.

- Le principal IAM qui a créé le cluster est le seul principal à pouvoir effectuer des appels au serveur d'API Kubernetes à l'aide de `kubectl` ou de la AWS Management Console. Si vous souhaitez que d'autres principaux IAM aient accès à votre cluster, vous devez les ajouter. Pour plus d'informations, consultez Autoriser l'accès aux Kubernetes API et Autorisations nécessaires.
- Déployez un exemple d'application sur votre cluster.
- Avant de déployer un cluster pour une utilisation en production, nous vous recommandons de vous familiariser avec tous les paramètres des clusters et des nœuds. Certains paramètres, tels que l'activation de l'accès SSH aux nœuds Amazon EC2, doivent être effectués lors de la création du cluster.
- Pour renforcer la sécurité de votre cluster, configurez le plugin Amazon VPC Container Networking Interface pour utiliser les rôles IAM pour les comptes de service.

Clusters Amazon EKS

Un cluster Amazon EKS comprend deux composants principaux :

- Le plan de contrôle Amazon EKS ;
- Les nœuds Amazon EKS enregistrés avec le plan de contrôle

Le plan de contrôle Amazon EKS se compose de nœuds de plan de contrôle qui exécutent le logiciel Kubernetes, comme `etcd` et le serveur d'API Kubernetes. Le plan de contrôle s'exécute dans un compte géré par AWS, et l'KubernetesAPI est exposée via le point de terminaison Amazon EKS associé à votre cluster. Chaque plan de contrôle de cluster Amazon EKS gère un seul client, est unique et s'exécute sur son propre ensemble d'instances Amazon EC2.

Toutes les données stockées par les `etcd` nœuds et les volumes Amazon EBS associés sont cryptées à l'aide AWS KMS de. Le plan de contrôle de cluster est mis en service sur plusieurs zones de disponibilité et placé en avant-plan par un Elastic Load Balancing Network Load Balancer. Amazon EKS alloue également des interfaces réseau Elastic dans vos sous-réseaux VPC, pour fournir une connectivité à partir des instances du plan de contrôle vers les nœuds (par exemple, pour prendre en charge des flux de données `proxy` des `logs` `kubectl exec`).

Important

Dans l'environnement Amazon EKS, le stockage `etcd` est limité à 8 Gio, conformément aux conseils donnés en amont. Vous pouvez surveiller une métrique pour la taille actuelle de la base de données en exécutant la commande suivante. Si votre cluster a une version Kubernetes inférieure à `1.28`, remplacez *`apiserver_storage_size_bytes`* par :

- Version `1.27` et `1.26` de Kubernetes : **`apiserver_storage_db_total_size_in_bytes`**
- Version `1.25` et inférieure de Kubernetes : **`etcd_db_total_size_in_bytes`**

```
kubectl get --raw=/metrics | grep "apiserver_storage_size_bytes"
```

Les nœuds Amazon EKS s'exécutent dans votre AWS compte et se connectent au plan de contrôle de votre cluster via le point de terminaison du serveur API et un fichier de certificat créé pour votre cluster.

> **Note**
>
> - Vous pouvez découvrir comment fonctionnent les différents composants d'Amazon EKS dans Mise en réseau d'Amazon EKS.
> - Pour les clusters connectés, consultez Amazon EKS Connector.

Rubriques

Création d'un cluster Amazon EKS

Cette rubrique fournit une vue d'ensemble des options disponibles et décrit les éléments à prendre en compte lorsque vous créez un cluster Amazon EKS. Si vous devez créer un cluster sur un AWS avant-poste, consultezClusters locaux pour Amazon EKS sur AWS Outposts. Si c'est la première fois que vous créez un cluster Amazon EKS, nous vous recommandons de suivre plutôt l'un de nos guides Démarrer avec Amazon EKS. Ces guides vous aident à créer un cluster par défaut simple sans développer toutes les options disponibles.

Prérequis

- Un VPC et des sous-réseaux existants qui répondent aux exigences Amazon EKS. Avant de déployer un cluster pour une utilisation en production, nous vous recommandons de bien connaître les exigences du VPC et du sous-réseau. Si vous n'avez pas de VPC ni de sous-réseaux, vous pouvez les créer à l'aide d'un modèle fourni par Amazon EKS. AWS CloudFormation
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- Un principal IAM disposant des autorisations `create` et `describe` sur un cluster Amazon EKS. Pour plus d'informations, consultez Créer un cluster Kubernetes local sur un Outpost et Affichage de la liste ou description de tous les clusters.

Pour créer un cluster Amazon EKS

1. Si vous avez déjà un rôle IAM de cluster, ou si vous allez créer votre cluster avec `eksctl`, vous pouvez ignorer cette étape. Par défaut, `eksctl` crée un rôle pour vous.

   Pour créer un rôle IAM de cluster Amazon EKS

   1. Exécutez la commande suivante pour créer un fichier JSON de politique d'approbation IAM.

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

2. Créez le rôle IAM de cluster Amazon EKS. Si nécessaire, faites précéder *eks-cluster-role-trust-policy.json* par le chemin d'accès sur votre ordinateur sur lequel vous avez enregistré le fichier lors de l'étape précédente. La commande associe la politique d'approbation que vous avez créée lors de l'étape précédente à ce rôle. Pour créer un rôle IAM, le principal IAM qui crée le rôle doit se voir attribuer l'action (autorisation) IAM `iam:CreateRole`.

```
aws iam create-role --role-name myAmazonEKSClusterRole --assume-role-policy-
document file://"eks-cluster-role-trust-policy.json"
```

3. Vous pouvez attribuer la politique gérée par Amazon EKS ou créer votre propre politique personnalisée. Pour connaître les autorisations minimales que vous devez utiliser dans votre politique personnalisée, consultez Rôle IAM de cluster Amazon EKS.

   Attachez la politique gérée par Amazon EKS appelée `AmazonEKSClusterPolicy` au rôle. Pour attacher une politique IAM à un principal IAM, le principal qui attache la politique doit se voir attribuer l'une des actions (autorisations) IAM `iam:AttachUserPolicy` ou `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name myAmazonEKSClusterRole
```

2. Créez un cluster Amazon EKS.

   Vous pouvez créer un cluster en utilisant le `eksctl` AWS Management Console, ou le AWS CLI.

   eksctl

   Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour créer votre cluster

Créez un cluster `IPv4` Amazon EKS avec la version de Kubernetes par défaut d'Amazon EKS dans votre Région AWS par défaut. Avant d'exécuter la commande, effectuez les remplacements suivants :

- *`region-code`*Remplacez-le par Région AWS celui dans lequel vous souhaitez créer votre cluster.
- Remplacez *`my-cluster`* par un nom pour votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
- Remplacez la version *`1.29`* par n'importe quelle version prise en charge par Amazon EKS.

  > **Note**
  >
  > Pour déployer un cluster 1.30 à ce stade, vous devez utiliser le AWS Management Console ou le AWS CLI.

- Modifiez les valeurs de `vpc-private-subnets` pour répondre à vos besoins. Vous pouvez également ajouter d'autres identifiants. Vous devez spécifier au moins deux ID de sous-réseau. Si vous préférez spécifier des sous-réseaux publics, vous pouvez remplacer `--vpc-private-subnets` par `--vpc-public-subnets`. Les sous-réseaux publics ont une table de routage associée avec un routage vers une passerelle Internet, mais les sous-réseaux privés n'ont pas de table de routage associée. Nous recommandons d'utiliser des sous-réseaux privés dans la mesure du possible.

  Les sous-réseaux que vous choisissez doivent respecter les exigences relatives aux sous-réseaux d'Amazon EKS. Avant de sélectionner des sous-réseaux, nous vous recommandons de bien connaître toutes les exigences et considérations requises pour Amazon EKS VPC et les sous-réseaux.

```
eksctl create cluster --name my-cluster --region region-code --version 1.29 --
vpc-private-subnets subnet-ExampleID1,subnet-ExampleID2 --without-nodegroup
```

L'approvisionnement de cluster dure plusieurs minutes. Pendant la création du cluster, plusieurs lignes de sortie apparaissent. La dernière ligne de sortie est similaire à celle de l'exemple suivant.

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> Pour afficher la plupart des options qui peuvent être spécifiées lors de la création d'un cluster avec `eksctl`, utilisez la commande **`eksctl create cluster --help`**. Pour consulter toutes les options disponibles, vous pouvez utiliser un fichier `config`. Pour plus d'informations, consultez Utilisation des fichiers de configuration et du schéma du fichier de configuration dans la documentation `eksctl`. Vous pouvez trouver des exemples de fichiers de configuration sur GitHub.

Paramètres facultatifs

Les paramètres suivants sont des paramètres facultatifs qui, si nécessaire, doivent être ajoutés à la commande précédente. Vous pouvez activer ces options uniquement lorsque vous créez le cluster, et non après. Si vous devez spécifier ces options, vous devez créer le cluster avec un fichier de configuration `eksctl` et spécifier les paramètres, plutôt que d'utiliser la commande précédente.

- Si vous souhaitez spécifier un ou plusieurs groupes de sécurité qu'Amazon EKS attribue aux interfaces réseau qu'il crée, spécifiez l'option `securityGroup`.

  Que vous choisissiez ou non un groupe de sécurité, Amazon EKS crée un groupe de sécurité qui permet la communication entre votre cluster et votre VPC. Amazon EKS associe ce groupe de sécurité, et tout ce que vous choisissez, aux interfaces réseau qu'il crée. Pour plus d'informations sur le groupe de sécurité de cluster créé par Amazon EKS, consultez the section called “Exigences de groupe de sécurité”. Vous pouvez modifier les règles dans le groupe de sécurité de cluster créé par Amazon EKS.

- Si vous souhaitez spécifier à partir de quel bloc de routage inter-domaines sans classe (CIDR) `IPv4` Kubernetes attribue des adresses IP de service, spécifiez l'option `serviceIPv4CIDR`.

  La spécification de votre propre plage peut aider à prévenir les conflits entre les services Kubernetes et d'autres réseaux appairés ou connectés à votre VPC. Entrez une plage en notation CIDR. Par exemple : `10.2.0.0/16`.

  Le bloc d'adresse CIDR doit répondre aux critères suivants :

  - Être situé dans l'une des plages suivantes : `10.0.0.0/8`, `172.16.0.0/12`, ou `192.168.0.0/16`.
  - Avoir une taille minimale de `/24` et une taille maximale de `/12`.
  - Ne pas chevaucher la plage du VPC pour vos ressources Amazon EKS.

  Vous ne pouvez spécifier cette option que lorsque vous utilisez la famille d'adresses `IPv4` et uniquement lors de la création du cluster. Si vous ne spécifiez pas cela, alors Kubernetes attribue des adresses IP de service à partir des blocs d'adresse CIDR `10.100.0.0/16` ou `172.20.0.0/16`.

- Si vous créez un cluster et que vous souhaitez que le cluster attribue des adresses `IPv6` addresses to Pods aux services au lieu d'adresses `IPv4` spécifiez l'option `ipFamily`.

  Par défaut, Kubernetes attribue des adresses `IPv4` aux Pods et services. Avant de décider d'utiliser la famille `IPv6`, assurez-vous que vous connaissez bien toutes les considérations et les exigences des rubriques the section called "Exigences et considérations requises pour le VPC", the section called "Exigences et considérations requises pour les sous-réseaux", the section called "Exigences de groupe de sécurité", et the section called "IPv6". Si vous choisissez la famille `IPv6`, vous ne pouvez pas spécifier une plage d'adresses pour que Kubernetes attribue des adresses de service `IPv6` comme vous pouvez le faire pour la famille `IPv4`. Kubernetes attribue des adresses de service à partir de la plage d'adresses locale unique (`fc00::/7`).

AWS Management Console

Pour créer votre cluster

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez Add cluster (Ajouter un cluster), puis choisissez Create (Créer).
3. Sur la page Configurer le cluster, renseignez les champs suivants :
   - Nom : nom de votre cluster. Le nom ne peut contenir que des caractères alphanumériques (en distinguant majuscules et minuscules), des traits d'union et des traits de soulignement. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
   - Version de Kubernetes : version de Kubernetes à utiliser pour votre cluster. Nous vous recommandons de sélectionner la dernière version, sauf si vous avez besoin d'une version antérieure.
   - Rôle de service de cluster : choisissez le rôle IAM du cluster Amazon EKS que vous avez créé pour permettre au plan Kubernetes de contrôle de gérer les AWS ressources en votre nom.
   - Chiffrement des secrets : (facultatif) choisissez d'activer le chiffrement des secrets Kubernetes à l'aide d'une clé KMS. Vous pouvez également activer cette option une fois que vous avez créé votre cluster. Avant d'activer cette fonctionnalité, assurez-vous que vous connaissez bien les informations de Activation du chiffrement du secret sur un cluster existant.
   - Identifications◦: (facultatif) ajoutez des identifications à votre cluster. Pour plus d'informations, consultez Étiquetage de vos ressources Amazon EKS.

     Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.
4. Sur la page Spécifier les réseaux sélectionnez des valeurs pour les champs suivants :
   - VPC : choisissez un VPC existant qui répond aux exigences relatives au VPC Amazon EKS pour y créer votre cluster. Avant de sélectionner un VPC, nous vous recommandons de bien connaître toutes les exigences et considérations de Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux. Vous ne pouvez pas modifier le VPC que vous souhaitez utiliser après la création d'un cluster. Si

aucun VPC n'est répertorié, vous devez d'abord en créer un. Pour plus d'informations, consultez Création d'un VPC pour votre cluster Amazon EKS.

- Sous-réseaux : par défaut, tous les sous-réseaux disponibles dans le VPC spécifié dans le champ précédent sont présélectionnés. Vous devez en sélectionner au moins deux.

  Les sous-réseaux que vous choisissez doivent respecter les exigences relatives aux sous-réseaux d'Amazon EKS. Avant de sélectionner des sous-réseaux, nous vous recommandons de bien connaître toutes les exigences et considérations requises pour Amazon EKS VPC et les sous-réseaux.

  Groupes de sécurité : (facultatif) spécifiez un ou plusieurs groupes de sécurité créant des interfaces réseau auxquelles vous souhaitez associer Amazon EKS.

  Que vous choisissiez ou non un groupe de sécurité, Amazon EKS crée un groupe de sécurité qui permet la communication entre votre cluster et votre VPC. Amazon EKS associe ce groupe de sécurité, et tout ce que vous choisissez, aux interfaces réseau qu'il crée. Pour plus d'informations sur le groupe de sécurité de cluster créé par Amazon EKS, consultez the section called “Exigences de groupe de sécurité”. Vous pouvez modifier les règles dans le groupe de sécurité de cluster créé par Amazon EKS.
- Choisissez la famille d'adresses IP du cluster : vous pouvez choisir entre IPv4 et IPv6.

  Par défaut, Kubernetes attribue des adresses `IPv4` aux Pods et services. Avant de décider d'utiliser la famille `IPv6`, assurez-vous que vous connaissez bien toutes les considérations et les exigences des rubriques the section called “Exigences et considérations requises pour le VPC”, the section called “Exigences et considérations requises pour les sous-réseaux”, the section called “Exigences de groupe de sécurité”, et the section called “IPv6”. Si vous choisissez la famille `IPv6`, vous ne pouvez pas spécifier une plage d'adresses pour que Kubernetes attribue des adresses de service `IPv6` comme vous pouvez le faire pour la famille `IPv4`. Kubernetes attribue des adresses de service à partir de la plage d'adresses locale unique (`fc00::/7`).

- (Facultatif) Choisissez Configurer la plage d'adresses IP du service Kubernetes et spécifiez une gamme de services **`IPv4`**.

  La spécification de votre propre plage peut aider à prévenir les conflits entre les services Kubernetes et d'autres réseaux appairés ou connectés à votre VPC. Entrez une plage en notation CIDR. Par exemple : `10.2.0.0/16`.

Le bloc d'adresse CIDR doit répondre aux critères suivants :

- Être situé dans l'une des plages suivantes : `10.0.0.0/8`, `172.16.0.0/12`, ou `192.168.0.0/16`.
- Avoir une taille minimale de `/24` et une taille maximale de `/12`.
- Ne pas chevaucher la plage du VPC pour vos ressources Amazon EKS.

Vous ne pouvez spécifier cette option que lorsque vous utilisez la famille d'adresses `IPv4` et uniquement lors de la création du cluster. Si vous ne spécifiez pas cela, alors Kubernetes attribue des adresses IP de service à partir des blocs d'adresse CIDR `10.100.0.0/16` ou `172.20.0.0/16`.

- Pour Accès au point de terminaison de cluster, sélectionnez une option. Une fois votre cluster créé, vous pouvez modifier cette option. Avant de sélectionner une option autre que par défaut, assurez-vous de vous familiariser avec les options et leurs implications. Pour plus d'informations, consultez Contrôle d'accès au point de terminaison du cluster Amazon EKS..

Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.

5. (Facultatif) Sur la page Configurer l'observabilité, choisissez quelles options de Métriques et de Journalisation du plan de contrôle doivent être activées. Par défaut, chaque type de journal est désactivé.
   - Pour plus d'informations sur l'option de métriques Prometheus, consultez Activation des métriques Prometheus lors de la création d'un cluster.
   - Pour plus d'informations sur les options de Journalisation du plan de contrôle, consultez Journalisation de plan de contrôle d'Amazon EKS.

   Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.
6. Sur la page Select add-ons (Sélectionner les modules complémentaires), choisissez les modules complémentaires que vous souhaitez ajouter à votre cluster. Vous pouvez choisir autant de modules complémentaires Amazon EKS et de modules complémentaires AWS Marketplace que vous le souhaitez. Si les modules complémentaires AWS Marketplace que vous souhaitez installer ne figurent pas dans la liste, vous pouvez rechercher les modules complémentaires AWS Marketplace disponibles en saisissant du texte dans la zone de recherche. Vous pouvez également effectuer une recherche par catégorie, fournisseur ou modèle de tarification, puis choisir les modules complémentaires

dans les résultats de la recherche. Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.

7. Sur la page Configurer les paramètres de modules complémentaires sélectionnés, sélectionnez la version que vous voulez installer. Vous pourrez toujours effectuer une mise à jour vers une version ultérieure après la création du cluster. Vous pourrez mettre à jour la configuration de chaque module complémentaire après la création du cluster. Pour plus d'informations sur la configuration des modules complémentaires, consultez Mise à jour d'un module complémentaire. Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.
8. Sur la page Vérifier et créer, passez en revue les informations que vous avez saisies ou sélectionnées sur les pages précédentes. Si vous devez apporter des modifications, choisissez Modifier. Quand vous êtes satisfait, choisissez Créer. Le champ État affiche EN COURS DE CRÉATION pendant que le cluster est provisionné.

   > **Note**
   >
   > Il est possible que vous receviez un message d'erreur indiquant que l'une des zones de disponibilité de votre demande ne dispose pas d'une capacité suffisante pour créer un cluster Amazon EKS. Si cela se produit, la sortie de l'erreur contient les zones de disponibilité qui peuvent prendre en charge un nouveau cluster. Essayez à nouveau de créer votre cluster avec au moins deux sous-réseaux situés dans les zones de disponibilité prises en charge pour votre compte. Pour plus d'informations, consultez Capacité insuffisante.

   L'approvisionnement de cluster dure plusieurs minutes.

AWS CLI

Pour créer votre cluster

1. Créez votre cluster à l'aide de la commande suivante. Avant d'exécuter la commande, effectuez les remplacements suivants :
   - *region-code*Remplacez-le par Région AWS celui dans lequel vous souhaitez créer votre cluster.

- Remplacez *my-cluster* par un nom pour votre cluster. Le nom ne peut contenir que des caractères alphanumériques (en distinguant majuscules et minuscules), des traits d'union et des traits de soulignement. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
- Remplacez la version *1.30* par n'importe quelle version prise en charge par Amazon EKS.
- Remplacez *111122223333* par l'ID de votre compte et *myAmazonEKSClusterRole* par le nom de votre rôle IAM de cluster.
- Remplacez les valeurs de `subnetIds` par vos propres valeurs. Vous pouvez également ajouter d'autres identifiants. Vous devez spécifier au moins deux ID de sous-réseau.

  Les sous-réseaux que vous choisissez doivent respecter les exigences relatives aux sous-réseaux d'Amazon EKS. Avant de sélectionner des sous-réseaux, nous vous recommandons de bien connaître toutes les exigences et considérations requises pour Amazon EKS VPC et les sous-réseaux.
- Si vous ne voulez pas spécifier un ID de groupe de sécurité, supprimez `,securityGroupIds=sg-`*ExampleID1* depuis la commande. Si vous souhaitez spécifier un ou plusieurs ID de groupe de sécurité, remplacez les valeurs de `securityGroupIds` par vos propres valeurs. Vous pouvez également ajouter d'autres identifiants.

  Que vous choisissiez ou non un groupe de sécurité, Amazon EKS crée un groupe de sécurité qui permet la communication entre votre cluster et votre VPC. Amazon EKS associe ce groupe de sécurité, et tout ce que vous choisissez, aux interfaces réseau qu'il crée. Pour plus d'informations sur le groupe de sécurité de cluster créé par Amazon EKS, consultez the section called “Exigences de groupe de sécurité”. Vous pouvez modifier les règles dans le groupe de sécurité de cluster créé par Amazon EKS.

```
aws eks create-cluster --region region-code --name my-cluster --kubernetes-
version 1.30 \
   --role-arn arn:aws:iam::111122223333:role/myAmazonEKSClusterRole \
   --resources-vpc-config
 subnetIds=subnet-ExampleID1,subnet-ExampleID2,securityGroupIds=sg-ExampleID1
```

> **Note**
>
> Il est possible que vous receviez un message d'erreur indiquant que l'une des zones de disponibilité de votre demande ne dispose pas d'une capacité suffisante pour créer un cluster Amazon EKS. Si cela se produit, la sortie de l'erreur contient les zones de disponibilité qui peuvent prendre en charge un nouveau cluster. Essayez à nouveau de créer votre cluster avec au moins deux sous-réseaux situés dans les zones de disponibilité prises en charge pour votre compte. Pour plus d'informations, consultez Capacité insuffisante.

Paramètres facultatifs

Les paramètres suivants sont des paramètres facultatifs qui, si nécessaire, doivent être ajoutés à la commande précédente. Vous pouvez activer ces options uniquement lorsque vous créez le cluster, et non après.

- Si vous souhaitez spécifier à partir de quel bloc de routage inter-domaines sans classe (CIDR) `IPv4` Kubernetes attribue des adresses IP de service, vous devez le spécifier en ajoutant le **`--kubernetes-network-config serviceIpv4Cidr=`*`CIDR block`*** à la commande suivante.

  La spécification de votre propre plage peut aider à prévenir les conflits entre les services Kubernetes et d'autres réseaux appairés ou connectés à votre VPC. Entrez une plage en notation CIDR. Par exemple : `10.2.0.0/16`.

  Le bloc d'adresse CIDR doit répondre aux critères suivants :

  - Être situé dans l'une des plages suivantes : `10.0.0.0/8`, `172.16.0.0/12`, ou `192.168.0.0/16`.
  - Avoir une taille minimale de `/24` et une taille maximale de `/12`.
  - Ne pas chevaucher la plage du VPC pour vos ressources Amazon EKS.

  Vous ne pouvez spécifier cette option que lorsque vous utilisez la famille d'adresses `IPv4` et uniquement lors de la création du cluster. Si vous ne spécifiez pas cela, alors Kubernetes attribue des adresses IP de service à partir des blocs d'adresse CIDR `10.100.0.0/16` ou `172.20.0.0/16`.

- Si vous créez un cluster et que vous souhaitez qu'il attribue des adresses IPv6 aux Pods et aux services au lieu d'adresses IPv4 ajoutez **`--kubernetes-network-config ipFamily=ipv6`** à la commande suivante.

  Par défaut, Kubernetes attribue des adresses IPv4 aux Pods et services. Avant de décider d'utiliser la famille IPv6, assurez-vous que vous connaissez bien toutes les considérations et les exigences des rubriques the section called "Exigences et considérations requises pour le VPC", the section called "Exigences et considérations requises pour les sous-réseaux", the section called "Exigences de groupe de sécurité", et the section called "IPv6". Si vous choisissez la famille IPv6, vous ne pouvez pas spécifier une plage d'adresses pour que Kubernetes attribue des adresses de service IPv6 comme vous pouvez le faire pour la famille IPv4. Kubernetes attribue des adresses de service à partir de la plage d'adresses locale unique (`fc00::/7`).

2. Le provisionnement du cluster prend quelques minutes. Vous pouvez vérifier le statut de votre cluster avec la commande suivante.

   ```
   aws eks describe-cluster --region region-code --name my-cluster --query
    "cluster.status"
   ```

   Ne passez pas à l'étape suivante tant que la sortie retournée n'est pas `ACTIVE`.

3. Si vous avez créé votre cluster à l'aide de `eksctl`, vous pouvez ignorer cette étape. Cela est dû au fait que `eksctl` a déjà terminé cette étape pour vous. Activez `kubectl` pour communiquer avec votre cluster en ajoutant un nouveau contexte au fichier `kubectl config`. Pour plus d'informations sur la création et la mise à jour du fichier, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   L'exemple qui suit illustre un résultat.

   ```
   Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
   username/.kube/config
   ```

4. Confirmez la communication avec votre cluster en exécutant la commande suivante.

   ```
   kubectl get svc
   ```

L'exemple qui suit illustre un résultat.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

5. (Recommandé) Pour utiliser certains modules complémentaires Amazon EKS ou pour permettre à des Kubernetes charges de travail individuelles de bénéficier d'autorisations AWS Identity and Access Management (IAM) spécifiques, créez un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Vous n'avez besoin de créer un fournisseur OIDC IAM pour votre cluster qu'une seule fois. Pour en savoir plus sur les modules complémentaires Amazon EKS, consultez Modules complémentaires Amazon EKS. Pour en savoir plus sur l'attribution des autorisations IAM spécifiques à vos applications, consultez Rôles IAM pour les comptes de service.
6. (Recommandé) Configurez votre cluster pour le plugin Amazon VPC CNI plugin for Kubernetes avant de déployer des nœuds Amazon EC2 sur votre cluster. Par défaut, le plugin a été installé avec votre cluster. Lorsque vous ajoutez des nœuds Amazon EC2 à votre cluster, le plugin est automatiquement déployé sur chaque nœud Amazon EC2 que vous ajoutez. Le plugin nécessite que vous attachiez l'une des politiques IAM suivantes à un rôle IAM :

   politique IAM gérée par **`AmazonEKS_CNI_Policy`**

   Si votre cluster utilise la famille `IPv4`

   Une politique IAM que vous créez

   Si votre cluster utilise la famille `IPv6`

   Le rôle IAM auquel vous attachez la politique peut être le rôle IAM du nœud ou un rôle dédié utilisé uniquement pour le plugin. Nous vous recommandons d'associer la politique à ce rôle. Pour plus d'informations sur la création du rôle, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA) ou Rôle IAM de nœud Amazon EKS.
7. Si vous avez déployé votre cluster à l'aide du AWS Management Console, vous pouvez ignorer cette étape. La AWS Management Console déploie le Amazon VPC CNI plugin for Kubernetes, CoreDNS, et des modules complémentaires Amazon EKS `kube-proxy`, par défaut.

   Si vous déployez votre cluster à l'aide de `eksctl` ou de la AWS CLI, le Amazon VPC CNI plugin for Kubernetes, CoreDNS, et des modules complémentaires autogérés `kube-proxy` sont déployés. Vous pouvez migrer le Amazon VPC CNI plugin for Kubernetes, CoreDNS, et

les modules complémentaires autogérés `kube-proxy` qui sont déployés avec votre cluster vers des modules complémentaires Amazon EKS. Pour plus d'informations, consultez Modules complémentaires Amazon EKS.

8. (Facultatif) Si vous ne l'avez pas encore fait, vous pouvez activer les métriques Prometheus pour votre cluster. Pour plus d'informations, consultez Création d'un scraper dans le Guide de l'utilisateur Amazon Managed Service for Prometheus.
9. Si vous avez activé les métriques Prometheus, vous devez configurer votre `aws-auth ConfigMap` pour donner au scraper des autorisations dans le cluster. Pour plus d'informations, consultez Configuration de votre cluster Amazon EKS dans le Guide de l'utilisateur Amazon Managed Service for Prometheus.
10. Si vous prévoyez de déployer des charges de travail sur votre cluster qui utilisent des volumes Amazon EBS et que vous avez créé un cluster `1.23` ou ultérieur, vous devez alors installer Pilote CSI Amazon EBS à votre cluster avant de déployer les charges de travail.

Étapes suivantes recommandées :

- Le principal IAM qui a créé le cluster est le seul principal à avoir accès au cluster. Accordez des autorisations à d'autres principaux IAM afin qu'ils puissent accéder à votre cluster.
- Si le principal IAM qui a créé le cluster ne dispose que des autorisations IAM minimales référencées dans les prérequis, vous pouvez alors ajouter des autorisations Amazon EKS supplémentaires pour ce principal. Pour plus d'informations sur l'octroi d'autorisations Amazon EKS aux principaux IAM, consultez la rubrique Gestion des identités et des accès pour Amazon EKS.
- Si vous souhaitez que le principal IAM qui a créé le cluster ou que n'importe quel autre principal affiche les ressources Kubernetes dans la console Amazon EKS, accordez les Autorisations nécessaires aux entités.
- Si vous souhaitez que les nœuds et les principaux IAM accèdent à votre cluster depuis votre VPC, activez le point de terminaison privé pour votre cluster. Le point de terminaison public est activé par défaut. Vous pouvez désactiver le point de terminaison public après avoir activé le point de terminaison privé, si vous le souhaitez. Pour plus d'informations, consultez Contrôle d'accès au point de terminaison du cluster Amazon EKS..
- Activer le chiffrement des secrets pour votre cluster.
- Configurer la journalisation de votre cluster.
- Ajouter des nœuds à votre cluster.

Informations sur le cluster

Les informations sur le cluster Amazon EKS fournit des recommandations pour vous aider à suivre les bonnes pratiques Amazon EKS et Kubernetes. Chaque cluster Amazon EKS est soumis à des contrôles automatiques et récurrents par rapport à une liste d'informations organisée par Amazon EKS. Ces contrôles d'informations sont entièrement gérés par Amazon EKS et proposent des recommandations sur la manière de traiter les éventuels résultats.

Utilisation recommandée des informations sur les clusters :

- Avant de mettre à jour Kubernetes la version de votre cluster, consultez les informations relatives au cluster dans la console EKS.
- Si votre cluster a identifié des problèmes, passez-les en revue et apportez les correctifs appropriés. Les problèmes incluent les liens vers Amazon EKS etKubernetes.
- Après avoir résolu les problèmes, attendez que les informations du cluster soient actualisées. Si tous les problèmes ont été résolus, mettez à jour votre cluster.

Actuellement, Amazon EKS renvoie uniquement des informations relatives à l'état de préparation de la mise à niveau des versions de Kubernetes.

Les informations sur les mises à niveau identifient les problèmes potentiels susceptibles d'avoir un impact sur les mises à niveau des cluster Kubernetes. Cela minimise les efforts que les administrateurs consacrent à la préparation des mises à niveau et augmente la fiabilité des applications sur les nouvelles versions de Kubernetes. Les clusters sont automatiquement analysés par Amazon EKS par rapport à une liste de problèmes susceptibles d'avoir un impact sur la mise à niveau de la version de Kubernetes. Amazon EKS met fréquemment à jour la liste des vérifications des informations en fonction des révisions des modifications apportées à chaque version publiée de Kubernetes.

Les informations de mise à niveau d'Amazon EKS accélèrent le processus de test et de vérification des nouvelles versions. Elles permettent également aux administrateurs de clusters et aux développeurs d'applications de tirer parti des fonctionnalités Kubernetes les plus récentes en mettant en évidence les problèmes et en proposant des conseils pour y remédier. Pour consulter la liste des vérifications d'informations effectuées et les éventuels problèmes pertinents identifiés par Amazon EKS, vous pouvez appeler l'API `ListInsights` Amazon EKS ou consulter la console Amazon EKS.

Les informations sur les clusters sont mises à jour régulièrement. Vous ne pouvez pas actualiser manuellement les informations du cluster. Si vous corrigez un problème de cluster, la mise à jour des informations sur le cluster prendra un certain temps. Pour déterminer si un correctif a réussi, comparez l'heure à laquelle la modification a été déployée à la « date de dernière actualisation » de l'aperçu du cluster.

Afficher les informations relatives au cluster (console)

Pour consulter les informations d'un cluster Amazon EKS :

a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

b. Dans la liste de clusters, choisissez le nom du cluster Amazon EKS pour lequel vous souhaitez obtenir des informations.

c. Choisissez l'onglet Mettre à niveau les informations.

d. Sur la page Mettre à niveau les informations, vous verrez les champs suivants :

- Nom : vérification effectuée par Amazon EKS par rapport au cluster.
- Statut des informations : une information dont le statut est « Erreur » signifie généralement que la version Kubernetes affectée est N+1 de la version actuelle du cluster, tandis qu'un statut « Avertissement » signifie que l'information s'applique à une future version Kubernetes N+2 ou plus. Une information dont le statut est « Réussi » signifie qu'Amazon EKS n'a détecté aucun problème associé à cette vérification d'informations dans votre cluster. Une information dont le statut est « Inconnu » signifie qu'Amazon EKS n'est pas en mesure de déterminer si votre cluster est concerné par cette vérification des informations.
- Version : la version Kubernetes vérifiée par l'analyse pour détecter d'éventuels problèmes.
- Heure de la dernière actualisation (UTC-5:00) : heure à laquelle le statut de l'information a été actualisé pour la dernière fois pour ce cluster.
- Heure de la dernière transition (UTC -5:00) : heure à laquelle le statut de cette information a été modifié pour la dernière fois.
- Description : informations provenant de la vérification des informations, qui incluent l'alerte et les mesures correctives recommandées.

Afficher les informations sur le cluster (AWS CLI)

Pour consulter les informations d'un cluster Amazon EKS :

a. Déterminez le cluster que vous souhaitez vérifier pour obtenir des informations. La commande suivante répertorie les information d'un cluster spécifique. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée :

- Remplacez *region-code* par le code de votre Région AWS.
- Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks list-insights --region region-code --cluster-name my-cluster
```

L'exemple qui suit illustre un résultat.

```
{
    "insights": [
        {
            "category": "UPGRADE_READINESS",
            "name": "Deprecated APIs removed in Kubernetes v1.29",
            "insightStatus": {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30 days."
            },
            "kubernetesVersion": "1.29",
            "lastTransitionTime": 1698774710.0,
            "lastRefreshTime": 1700157422.0,
            "id": "123e4567-e89b-42d3-a456-579642341238",
            "description": "Checks for usage of deprecated APIs that are scheduled
 for removal in Kubernetes v1.29. Upgrading your cluster before migrating to the
 updated APIs supported by v1.29 could cause application impact."
        }
    ]
}
```

b. Pour obtenir une description des informations, exécutez la commande suivante. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée :

- Remplacez *region-code* par le code de votre Région AWS.
- Remplacez *123e4567-e89b-42d3-a456-579642341238* par l'ID d'information extrait de la liste des informations du cluster.

- Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-insight --region region-code --id 123e4567-e89b-42d3-
a456-579642341238 --cluster-name my-cluster
```

L'exemple qui suit illustre un résultat.

```
{
    "insight": {
        "category": "UPGRADE_READINESS",
        "additionalInfo": {
            "EKS update cluster documentation": "https://docs.aws.amazon.com/eks/
latest/userguide/update-cluster.html",
            "Kubernetes v1.29 deprecation guide": "https://kubernetes.io/docs/
reference/using-api/deprecation-guide/#v1-29"
        },
        "name": "Deprecated APIs removed in Kubernetes v1.29",
        "insightStatus": {
            "status": "PASSING",
            "reason": "No deprecated API usage detected within the last 30 days."
        },
        "kubernetesVersion": "1.29",
        "recommendation": "Update manifests and API clients to use newer Kubernetes
 APIs if applicable before upgrading to Kubernetes v1.29.",
        "lastTransitionTime": 1698774710.0,
        "lastRefreshTime": 1700157422.0,
        "categorySpecificSummary": {
            "deprecationDetails": [
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
flowschemas",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
flowschemas",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                },
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
prioritylevelconfigurations",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
prioritylevelconfigurations",
```

```
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                }
            ]
        },
        "id": "f6a11fe4-77f7-48c6-8326-9a13f022ecb3",
        "resources": [],
        "description": "Checks for usage of deprecated APIs that are scheduled for
 removal in Kubernetes v1.29. Upgrading your cluster before migrating to the updated
 APIs supported by v1.29 could cause application impact."
    }
}
```

Mise à jour d'une version Kubernetes de cluster Amazon EKS

Dès qu'une nouvelle version Kubernetes est disponible dans Amazon EKS, vous pouvez mettre à jour votre cluster Amazon EKS vers la version la plus récente.

> **Important**
>
> Une fois que vous avez mis à niveau un cluster, vous ne pouvez pas le rétrograder vers une version antérieure. Nous vous recommandons, avant de mettre à jour vers une nouvelle version de Kubernetes, de consulter les informations dans Versions Kubernetes Amazon EKS et également de consulter les étapes de mise à jour de cette rubrique.

Les nouvelles versions de Kubernetes introduisent parfois des modifications importantes. Nous vous recommandons donc de tester le comportement de vos applications par rapport à la nouvelle version de Kubernetes avant de procéder à la mise à jour sur vos clusters de production. Pour ce faire, créez un flux d'intégration continue afin de tester le comportement de votre application avant de passer à une nouvelle version Kubernetes.

Dans le processus de mise à jour, Amazon EKS lance de nouveaux nœuds de serveur d'API avec la version de Kubernetes mise à jour pour remplacer les versions existantes. Amazon EKS effectue des surveillances de l'état de disponibilité et de l'infrastructure standard pour le trafic réseau sur ces nouveaux nœuds pour vérifier qu'ils fonctionnent comme prévu. Cependant, une fois que vous avez commencé la mise à niveau du cluster, vous ne pouvez ni la suspendre ni l'arrêter. Si l'une de ces vérifications échoue, Amazon EKS annule le déploiement de l'infrastructure, et votre cluster reste

dans la version Kubernetes précédente. Les applications en cours d'exécution ne sont pas affectées, et votre cluster n'est jamais laissé dans un état non déterministe ou irrécupérable. Amazon EKS sauvegarde régulièrement tous les clusters gérés, et des mécanismes existent pour récupérer des clusters si nécessaire. Nous évaluons et améliorons en permanence nos processus de gestion de l'infrastructure Kubernetes.

Pour mettre à jour le cluster, Amazon EKS requiert jusqu'à cinq adresses IP disponibles correspondant aux sous-réseaux que vous avez spécifiés lors de la création de votre cluster. Amazon EKS crée de nouvelles interfaces réseau élastiques de cluster (interfaces réseau) dans l'un des sous-réseaux que vous avez spécifiés. Les interfaces réseau peuvent être créées dans des sous-réseaux différents de ceux dans lesquels se trouvent vos interfaces réseau existantes. Assurez-vous donc que les règles de votre groupe de sécurité autorisent la communication de cluster requise pour tous les sous-réseaux que vous avez spécifiés lors de la création de votre cluster. Si l'un des sous-réseaux que vous avez spécifiés lors de la création du cluster n'existe pas, ne dispose pas de suffisamment d'adresses IP disponibles ou de règles de groupe de sécurité permettant la communication nécessaire avec le cluster, la mise à jour peut échouer.

> **Note**
>
> Afin d'assurer une disponibilité constante du point de terminaison du serveur d'API de votre cluster, Amazon EKS déploie un plan de contrôle Kubernetes hautement disponible et effectue des mises à jour continues des instances du serveur d'API pendant les opérations de mise à jour. Afin de tenir compte des modifications d'adresses IP des instances de serveurs d'API qui prennent en charge votre point de terminaison de serveur d'API Kubernetes, il est important de vous assurer que les clients de votre serveur d'API gèrent les reconnexions de manière efficace. Les versions récentes de `kubectl` et les bibliothèques clientes Kubernetes officiellement prises en charge effectuent ce processus de reconnexion de manière transparente.

Mise à jour de la version Kubernetes de votre cluster Amazon EKS

Pour mettre à jour la version Kubernetes de votre cluster

1. Comparez la version Kubernetes de votre plan de contrôle de cluster à la version Kubernetes de vos nœuds.

   - Obtenez la version Kubernetes du plan de contrôle de votre cluster.

```
kubectl version
```

- Obtenez la version Kubernetes de vos nœuds. Cette commande renvoie tous les nœuds Amazon EC2 et Fargate autogérés et gérés. Chaque Pod Fargate est répertorié comme son propre nœud.

```
kubectl get nodes
```

Avant de mettre à jour votre plan de contrôle vers une nouvelle version de Kubernetes, assurez-vous que la version mineure de Kubernetes des nœuds gérés et des nœuds Fargate de votre cluster est identique à la version de votre plan de contrôle. Par exemple, si votre plan de contrôle exécute la version `1.29` et que l'un de vos nœuds exécute la version`1.28`, vous devez mettre à jour vos nœuds vers la version 1.30 `1.29` avant de mettre à jour votre plan de contrôle vers la version 1.30. Nous vous recommandons également de mettre à jour vos nœuds autogérés vers la même version que votre plan de contrôle avant de mettre à jour le plan de contrôle. Pour plus d'informations, consultez Mise à jour d'un groupe de nœuds gérés et Mises à jour des nœuds autogérés. Si vous avez des nœuds Fargate dont la version mineure est inférieure à la version du plan de contrôle, supprimez d'abord le Pod représenté par le nœud. Mettez ensuite à jour votre plan de contrôle. Tous les Pods restants seront mis à jour vers la nouvelle version une fois que vous les aurez redéployés.

2. Si la version de Kubernetes avec laquelle vous avez initialement déployé votre cluster était Kubernetes version `1.25` ou ultérieure, ignorez cette étape.

   Par défaut, le contrôleur d'admission de politique de sécurité du Pod est activé sur les clusters Amazon EKS. Avant de mettre à jour votre cluster, assurez-vous que des politiques de sécurité de Pod appropriées sont en place. Il s'agit ainsi d'éviter les problèmes de sécurité potentiels. Vous pouvez vérifier la politique par défaut à l'aide de la commande **`kubectl get psp eks.privileged`**.

   ```
   kubectl get psp eks.privileged
   ```

   Si vous recevez l'erreur suivante, veuillez consulter Stratégie de sécurité de Pod Amazon EKS par défaut avant de continuer.

```
Error from server (NotFound): podsecuritypolicies.extensions "eks.privileged" not
 found
```

3. Si la version de Kubernetes avec laquelle vous avez initialement déployé votre cluster était Kubernetes version `1.18` ou ultérieure, ignorez cette étape.

   Vous devrez peut-être supprimer un terme abandonné de votre manifeste CoreDNS.

   a. Vérifiez si votre manifeste CoreDNS a une ligne qui n'a que le mot `upstream`.

   ```
   kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
    grep upstream
   ```

   Si aucune sortie n'est renvoyée, cela signifie que votre manifeste n'a pas la ligne. Dans ce cas, passez à l'étape suivante. Si le mot `upstream` est retourné, supprimez la ligne.

   b. Supprimez la ligne près du haut du fichier qui ne contient que le mot `upstream` dans le fichier configmap. Ne changez rien d'autre dans le fichier. Une fois la ligne supprimée, enregistrez les modifications.

   ```
   kubectl edit configmap coredns -n kube-system -o yaml
   ```

4. Mettez à jour votre cluster à l'aide du `eksctl` AWS Management Console, ou du AWS CLI.

   Important

   - Si vous effectuez une mise à jour vers la version `1.23` et utilisez des volumes Amazon EBS dans votre cluster, alors vous devez installer le pilote Amazon EBS CSI dans votre cluster avant de mettre celui-ci à jour vers la version `1.23` afin d'éviter les perturbations liées à la charge de travail. Pour plus d'informations, consultez Kubernetes1,23 et Pilote CSI Amazon EBS.
   - Kubernetes `1.24` et versions ultérieures utilisent `containerd` comme environnement d'exécution du conteneur par défaut. Si vous passez à l'exécution `containerd` et que vous avez déjà configuré Fluentd pour Container Insights, vous devez effectuer la migration de Fluentd vers Fluent Bit avant de mettre à jour votre cluster. Les analyseurs Fluentd sont configurés pour analyser uniquement les messages du journal au format JSON. Contrairement à `dockerd`, l'exécution du conteneur `containerd` a des messages de journal qui ne sont pas au format JSON. Si vous ne migrez pas

vers Fluent Bit, certains des analyseurs Fluentd's configurés généreront un grand nombre d'erreurs à l'intérieur du conteneur Fluentd. Pour plus d'informations sur la migration, voir Configurer en Fluent Bit tant que DaemonSet pour envoyer des CloudWatch journaux vers Logs.

- Dans la mesure où Amazon EKS exécute un plan de contrôle hautement disponible, vous devez mettre à jour une seule version mineure à la fois. Pour plus d'informations sur cette exigence, consultez Politique de prise en charge des versions et versions asymétriques de Kubernetes. Supposez que la version actuelle de votre cluster soit la version `1.28` et que vous souhaitiez la mettre à jour vers la version `1.30`. Vous devez d'abord mettre à jour la version `1.28` de votre cluster vers la version `1.29`, puis mettre à jour la version `1.29` de votre cluster vers la version `1.30`.
- Vérifiez l'asymétrie de version entre Kubernetes `kube-apiserver` et `kubelet` sur vos nœuds.
  - À partir de la version `1.28` de Kubernetes, `kubelet` peut avoir jusqu'à trois versions mineures antérieures à `kube-apiserver`. Voir la politique d'asymétrie des versions en amont de Kubernetes.
  - Si la version de Kubernetes du `kubelet` de vos nœuds gérés et nœuds Fargate est `1.25` ou une version plus récente, vous pouvez mettre à jour votre cluster jusqu'à trois versions à l'avance sans mettre à jour la version `kubelet`. Par exemple, si la version du `kubelet` est `1.25`, vous pouvez mettre à jour la version de votre cluster Amazon EKS de `1.25` vers `1.26` ou vers `1.27` et vers `1.28` alors que le `kubelet` reste sur la version `1.25`.
  - Si la version de Kubernetes du `kubelet` de vos nœuds gérés et nœuds Fargate est `1.24` ou une version plus ancienne, il ne peut être que sur maximum deux versions mineures plus anciennes que le `kube-apiserver`. En d'autres termes, si la version du `kubelet` est `1.24` ou une version antérieure, vous ne pouvez mettre à jour votre cluster que sur deux versions plus récentes. Par exemple, si la version du `kubelet` est `1.21`, vous pouvez mettre à jour la version de votre cluster Amazon EKS de `1.21` vers `1.22` et vers `1.23`, mais vous ne pouvez pas mettre à jour le cluster vers `1.24` alors que la version du `kubelet` reste sur `1.21`.
- Avant de commencer une mise à jour, il est recommandé de vérifier que le `kubelet` sur vos nœuds est identique à celle de la version Kubernetes de votre plan de contrôle.
- Si votre cluster est configuré avec une version d' Amazon VPC CNI plugin for Kubernetes antérieure à la version `1.8.0`, nous vous recommandons de mettre à jour

le plugin à la dernière version avant de mettre à jour votre cluster. Pour mettre à jour le plugin, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

- Si vous mettez à jour votre cluster vers la version `1.25` ou vers une version ultérieure et que vous avez déployé le AWS Load Balancer Controller dans votre cluster, mettez à jour le contrôleur vers la version `2.4.7` ou vers une version ultérieure avant de mettre à jour votre cluster vers la version `1.25`. Pour plus d'informations, consultez les notes de mise à jour de Kubernetes1,25.

eksctl

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation et de mise à jour de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

Mettez à jour la version Kubernetes de votre plan de contrôle Amazon EKS. Remplacez *my-cluster* par le nom de votre cluster. Remplacez *1.30* par le numéro de version pris en charge par Amazon EKS vers lequel vous souhaitez mettre à jour votre cluster. Pour obtenir une liste des numéros de version pris en charge, consultez Versions Kubernetes Amazon EKS.

```
eksctl upgrade cluster --name my-cluster --version 1.30 --approve
```

Cette mise à jour peut prendre plusieurs minutes.

AWS Management Console

a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

b. Sélectionnez le nom du cluster Amazon EKS à mettre à jour, puis choisissez Update cluster version (Mettre à jour la version du cluster).

c. Dans Version Kubernetes, sélectionnez la version vers laquelle mettre à jour votre cluster, puis sélectionnez Mettre à jour.

d. Dans Cluster name (Nom du cluster), saisissez le nom de votre cluster, puis sélectionnez Confirm (Confirmer).

   Cette mise à jour peut prendre plusieurs minutes.

AWS CLI

a. Mettez à jour votre cluster Amazon EKS à l'aide de la commande AWS CLI suivante. Remplacez les *example values* par vos propres valeurs. Remplacez *1.30* par le numéro de version pris en charge par Amazon EKS vers lequel vous souhaitez mettre à jour votre cluster. Pour obtenir une liste des numéros de version pris en charge, consultez Versions Kubernetes Amazon EKS.

```
aws eks update-cluster-version --region region-code --name my-cluster --
kubernetes-version 1.30
```

   L'exemple qui suit illustre un résultat.

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.30"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

b. Contrôlez l'état de mise à jour de votre cluster avec la commande suivante. Utilisez le nom de cluster et l'ID de mise à jour renvoyé par la commande précédente. Lorsqu'un

état `Successful` s'affiche, la mise à jour est terminée. Cette mise à jour peut prendre plusieurs minutes.

```
aws eks describe-update --region region-code --name my-cluster --update-
id b5f0ba18-9a87-4450-b5a0-825e6e84496f
```

L'exemple qui suit illustre un résultat.

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "Successful",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.30"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

5. Une fois la mise à jour de votre cluster terminée, mettez à jour vos nœuds vers la même version mineure Kubernetes que celle de votre cluster mis à jour. Pour plus d'informations, consultez Mises à jour des nœuds autogérés et Mise à jour d'un groupe de nœuds gérés. Tous les nouveaux Pods lancés sur Fargate ont une version `kubelet` correspondant à la version de votre cluster. Les Pods Fargate existants ne sont pas modifiés.
6. (Facultatif) Si vous avez déployé Kubernetes Cluster Autoscaler sur votre cluster avant de mettre à jour le cluster, mettez ce composant à jour vers la version la plus récente correspondant à la version majeure et mineure de Kubernetes vers laquelle vous avez effectué la mise à jour.

   a. Ouvrez la page des versions de Cluster Autoscaler dans un navigateur web et recherchez la dernière version de Cluster Autoscaler qui correspond aux versions majeure et mineure Kubernetes de votre cluster. Par exemple, si la version Kubernetes de votre cluster est

`1.30`, recherchez la version la plus récente du Cluster Autoscaler qui commence par `1.30`. Enregistrez le numéro de version sémantique (`1.30.n`, par exemple) de cette version à utiliser à l'étape suivante.

b. Identifiez l'image Cluster Autoscaler sur la version que vous avez enregistrée à l'étape précédente avec la commande suivante. Si nécessaire, remplacez *`1.30.n`* par votre propre valeur.

```
kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-
autoscaler=registry.k8s.io/autoscaling/cluster-autoscaler:v1.30.n
```

7. (Clusters avec des nœuds GPU uniquement) Si votre cluster comporte des groupes de nœuds avec prise en charge de GPU (par exemple `p3.2xlarge`), vous devez mettre à jour le contrôleur DaemonSet du plugin de périphérique NVIDIA pour Kubernetes sur votre cluster. Remplacez *`vX.X.X`* par la version NVIDIA/k8s-device-plugin souhaitée avant d'exécuter la commande suivante.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
nvidia-device-plugin.yml
```

8. Mettez à jour les modules complémentaires Amazon VPC CNI plugin for Kubernetes, CoreDNS et `kube-proxy`. Nous vous recommandons de mettre à jour les modules complémentaires avec les versions minimales répertoriées dans les jetons de compte de service.

   - Si vous utilisez des modules complémentaires Amazon EKS, dans la console Amazon EKS, sélectionnez Clusters, puis le nom du cluster que vous avez mis à jour dans le volet gauche. Les notifications s'affichent dans la console. Elles vous informent qu'une nouvelle version est disponible pour chaque module complémentaire disposant d'une mise à jour disponible. Pour mettre à jour un module complémentaire, sélectionnez l'onglet Add-ons (Modules complémentaires). Dans l'une des zones d'un module complémentaire dont une mise à jour est disponible, sélectionnez Update now (Mettre à jour maintenant), sélectionnez une version disponible, puis cliquez sur Update (Mise à jour).
   - Vous pouvez également utiliser le AWS CLI ou `eksctl` pour mettre à jour les modules complémentaires. Pour plus d'informations, consultez Mise à jour d'un module complémentaire.

9. Si nécessaire, mettez à jour votre version de `kubectl`. Vous devez utiliser une version `kubectl` qui se situe à une différence de version mineure près du plan de contrôle de votre cluster Amazon EKS. Par exemple, un client `kubectl 1.29` doit utiliser des clusters

Kubernetes, `1.28`, `1.29` et `1.30`. Vous pouvez vérifier votre version installée à l'aide de la commande suivante.

```
kubectl version --client
```

Suppression d'un cluster Amazon EKS

Lorsque vous avez terminé d'utiliser un cluster Amazon EKS, vous devez supprimer les ressources qui lui sont associées afin de ne pas entraîner de coûts superflus.

Pour supprimer un cluster connecté, reportez-vous à la section Annulation de l'enregistrement d'un cluster

Important

- Si des services actifs dans votre cluster sont associés à un équilibreur de charge, vous devez supprimer ces services avant de supprimer le cluster, afin que les équilibreurs de charge soient supprimés correctement. Dans le cas contraire, vous risquez d'avoir des ressources orphelines dans votre VPC, qui vous empêcheront de supprimer le VPC.
- Si vous recevez une erreur en raison de la suppression du créateur du cluster, reportez-vous à cet article pour la résoudre.
- Les ressources Amazon Managed Service for Prometheus ne font pas partie du cycle de vie du cluster et doivent être gérées indépendamment du cluster. Lorsque vous supprimez votre cluster, assurez-vous de supprimer également tous les scrapers applicables afin de mettre fin aux coûts applicables. Pour plus d'informations, consultez la section Rechercher et supprimer des scrapers dans le guide de l'utilisateur d'Amazon Managed Service for Prometheus.

Vous pouvez supprimer un cluster avec`eksctl`, le AWS Management Console, ou le AWS CLI.

eksctl

Pour supprimer un cluster Amazon EKS et des nœuds avec **`eksctl`**.

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

1. Répertoriez tous les services qui s'exécutent dans votre cluster.

   ```
   kubectl get svc --all-namespaces
   ```

2. Supprimez tous les services qui ont une valeur `EXTERNAL-IP` associée. Ces services ont un équilibreur de charge Elastic Load Balancing en avant-plan, et vous devez les supprimer dans Kubernetes pour que l'équilibreur de charge et les ressources associées soient correctement libérés.

   ```
   kubectl delete svc service-name
   ```

3. Supprimez le cluster et les nœuds qui y sont associés avec la commande suivante, en remplaçant *prod* par le nom de votre cluster.

   ```
   eksctl delete cluster --name prod
   ```

   Sortie :

   ```
   [#]  using region region-code
   [#]  deleting EKS cluster "prod"
   [#]  will delete stack "eksctl-prod-nodegroup-standard-nodes"
   [#]  waiting for stack "eksctl-prod-nodegroup-standard-nodes" to get deleted
   [#]  will delete stack "eksctl-prod-cluster"
   [#]  the following EKS cluster resource(s) for "prod" will be deleted: cluster.
    If in doubt, check CloudFormation console
   ```

AWS Management Console

Pour supprimer un cluster Amazon EKS à l'aide du AWS Management Console

1. Répertoriez tous les services qui s'exécutent dans votre cluster.

   ```
   kubectl get svc --all-namespaces
   ```

2. Supprimez tous les services qui ont une valeur `EXTERNAL-IP` associée. Ces services ont un équilibreur de charge Elastic Load Balancing en avant-plan, et vous devez les supprimer dans Kubernetes pour que l'équilibreur de charge et les ressources associées soient correctement libérés.

```
kubectl delete svc service-name
```

3. Supprimez tous les groupes de nœuds et les profils Fargate.

   a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

   b. Dans le panneau de navigation de gauche, sélectionnez Clusters Amazon EKS, puis dans la liste à onglets des clusters, sélectionnez le nom du cluster que vous voulez supprimer.

   c. Sélectionnez l'onglet Calcul et choisissez un groupe de nœuds à supprimer. Sélectionnez Supprimer, saisissez le nom du groupe de nœuds, puis sélectionnez Supprimer. Supprimez tous les groupes de nœuds du cluster.

   > **Note**
   >
   > Tous les groupes de nœuds répertoriés sont des groupes de nœuds gérés.

   d. Sélectionnez un profil Fargate à supprimer, choisissez Supprimer, saisissez le nom du profil, puis sélectionnez Supprimer. Supprimez tous les profils Fargate dans le cluster.

4. Supprimez toutes les piles de nœuds AWS CloudFormation autogérées.

   a. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

   b. Sélectionnez la pile du nœud à supprimer, puis choisissez Supprimer.

   c. Dans la boîte de dialogue de confirmation Delete stack (Supprimer la pile), choisissez Delete stack (Supprimer la pile). Supprimez toutes les piles de nœuds autogérées dans le cluster.

5. Supprimez le cluster.

   a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

   b. Sélectionnez le cluster à supprimer, puis choisissez Supprimer.

c. Dans l'écran de confirmation de suppression du cluster, choisissez Delete (Supprimer).

6. (Facultatif) Supprimez la pile VPC. AWS CloudFormation

   a. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

   b. Sélectionnez la pile VPC à supprimer, puis choisissez Delete (Supprimer).

   c. Dans la boîte de dialogue de confirmation Delete stack (Supprimer la pile), choisissez Delete stack (Supprimer la pile).

AWS CLI

Pour supprimer un cluster Amazon EKS à l'aide du AWS CLI

1. Répertoriez tous les services qui s'exécutent dans votre cluster.

```
kubectl get svc --all-namespaces
```

2. Supprimez tous les services qui ont une valeur `EXTERNAL-IP` associée. Ces services ont un équilibreur de charge Elastic Load Balancing en avant-plan, et vous devez les supprimer dans Kubernetes pour que l'équilibreur de charge et les ressources associées soient correctement libérés.

```
kubectl delete svc service-name
```

3. Supprimez tous les groupes de nœuds et les profils Fargate.

   a. Répertoriez les groupes de nœuds de votre cluster à l'aide de la commande suivante.

```
aws eks list-nodegroups --cluster-name my-cluster
```

> **Note**
>
> Tous les groupes de nœuds répertoriés sont des groupes de nœuds gérés.

   b. Supprimez chaque groupe de nœuds à l'aide de la commande suivante. Supprimez tous les groupes de nœuds du cluster.

```
aws eks delete-nodegroup --nodegroup-name my-nodegroup --cluster-name my-cluster
```

c. Répétez les profils Fargate dans votre cluster à l'aide de la commande suivante.

```
aws eks list-fargate-profiles --cluster-name my-cluster
```

d. Supprimez chaque profil Fargate avec la commande suivante. Supprimez tous les profils Fargate dans le cluster.

```
aws eks delete-fargate-profile --fargate-profile-name my-fargate-profile --cluster-name my-cluster
```

4. Supprimez toutes les piles de nœuds AWS CloudFormation autogérées.

a. Répertoriez vos AWS CloudFormation piles disponibles à l'aide de la commande suivante. Identifiez le nom du modèle du nœud dans la sortie obtenue.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

b. Supprimez chaque pile de nœud avec la commande suivante, en remplaçant *node-stack* par le nom de votre pile de nœuds. Supprimez toutes les piles de nœuds autogérées dans le cluster.

```
aws cloudformation delete-stack --stack-name node-stack
```

5. Supprimez le cluster avec la commande suivante, en remplaçant *my-cluster* par le nom de votre cluster.

```
aws eks delete-cluster --name my-cluster
```

6. (Facultatif) Supprimez la pile VPC. AWS CloudFormation

a. Répertoriez vos AWS CloudFormation piles disponibles à l'aide de la commande suivante. Recherchez le nom du modèle VPC dans la sortie obtenue.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

b. Supprimez la pile VPC avec la commande suivante, en remplaçant *my-vpc-stack* par le nom de votre pile VPC.

```
aws cloudformation delete-stack --stack-name my-vpc-stack
```

Contrôle d'accès au point de terminaison du cluster Amazon EKS.

Cette rubrique vous aide à activer l'accès privé pour le point de terminaison du serveur d'API Kubernetes de votre cluster Amazon EKS et à limiter ou désactiver complètement l'accès public depuis Internet.

Lorsque vous créez un cluster, Amazon EKS crée un point de terminaison pour le serveur d'API Kubernetes géré que vous utilisez pour communiquer avec votre cluster (à l'aide d'outils de gestion Kubernetes tels que `kubectl`). Par défaut, ce point de terminaison du serveur d'API est public sur Internet, et l'accès au serveur d'API est sécurisé à l'aide d'une combinaison de AWS Identity and Access Management (IAM) et de contrôle d'accès basé sur les Kubernetes rôles (RBAC) natif.

Vous pouvez activer l'accès privé au serveur d'API Kubernetes pour que toutes les communications entre vos nœuds et le serveur d'API restent au sein de votre VPC. Vous pouvez limiter les adresses IP qui peuvent accéder à votre serveur API à partir d'Internet, ou désactiver complètement l'accès Internet au serveur d'API.

> **Note**
>
> Comme ce point de terminaison est destiné au serveur d'KubernetesAPI et n'est pas un point de AWS PrivateLink terminaison traditionnel pour communiquer avec une AWS API, il n'apparaît pas en tant que point de terminaison dans la console Amazon VPC.

Lorsque vous activez l'accès privé au point de terminaison pour votre cluster, Amazon EKS crée une zone hébergée privée Route 53 en votre nom et l'associe au VPC de votre cluster. Cette zone hébergée privée est gérée par Amazon EKS et ne s'affiche pas dans les ressources Route 53 de votre compte. Pour que la zone hébergée privée achemine correctement le trafic vers votre serveur d'API, votre VPC doit avoir `enableDnsHostnames` et `enableDnsSupport` définis sur `true`, et les options DHCP définies pour votre VPC doivent inclure `AmazonProvidedDNS` dans leur liste de serveurs de nom de domaine. Pour plus d'informations, consultez Mise à jour du support DNS pour votre VPC dans le Guide de l'utilisateur d'Amazon VPC.

Vous pouvez définir les exigences d'accès au point de terminaison de votre serveur d'API lorsque vous créez un nouveau cluster, et vous pouvez mettre à jour l'accès au point de terminaison du serveur d'API pour un cluster à tout moment.

Modification de l'accès au point de terminaison de cluster

Utilisez les procédures de cette section afin de modifier l'accès au point de terminaison pour un cluster existant. Le tableau suivant présente les combinaisons d'accès au point de terminaison de serveur d'API prises en charge et leur comportement.

Options d'accès au point de terminaison du serveur d'API

Accès public au point de terminaison	Accès privé au point de terminaison	Attitude
Activé	Désactivées	• Il s'agit du comportement par défaut pour les nouveaux clusters Amazon EKS. • Les demandes d'API Kubernetes en provenance du VPC de votre cluster (comme pour la communication entre le nœud et le plan de contrôle) quittent le VPC, mais pas le réseau Amazon. • Le serveur d'API de votre cluster est accessible depuis internet. Si nécessaire, vous pouvez limiter les blocs CIDR qui peuvent accéder au point de terminaison public. Si vous limitez l'accès à des blocs CIDR spécifiques, il est recommandé d'activer également le point de terminaison privé ou de

Accès public au point de terminaison	Accès privé au point de terminaison	Attitude
		vous assurer que les blocs CIDR que vous spécifiez incluent les adresses auxquelles les nœuds et les Pods Fargate (si vous les utilisez) accèdent au point de terminaison public.
Activées	Activées	• Les demandes d'API Kubernetes dans le VPC de votre cluster (comme pour la communication entre le nœud et le plan de contrôle) utilisent le point de terminaison d'un VPC privé. • Le serveur d'API de votre cluster est accessible depuis internet. Si nécessaire, vous pouvez limiter les blocs CIDR qui peuvent accéder au point de terminaison public.

Accès public au point de terminaison	Accès privé au point de terminaison	Attitude
Désactivées	Activées	• Tout le trafic vers le serveur d'API de votre cluster doit provenir du VPC ou d'un réseau connecté de votre cluster. • Il n'y a pas d'accès public vers votre serveur d'API depuis Internet. Toutes les commandes `kubectl` doivent provenir du VPC ou d'un réseau connecté. Pour les options de connectivité, consultez Accès à un serveur d'API privé uniquement. • Le point de terminaison du serveur API du cluster est résolu par les serveurs DNS publics à une adresse IP privée à partir du VPC. Dans le passé, le point de terminaison ne pouvait être résolu qu'à partir du VPC. Si votre point de terminaison ne se résout pas à une adresse IP privée dans le VPC pour un cluster existant, vous pouvez : • Activer l'accès public, puis le désactiver à nouveau. Vous n'avez besoin de le faire qu'une seule

Accès public au point de terminaison	Accès privé au point de terminaison	Attitude
		fois pour un cluster et le point de terminaison se résoudra en une adresse IP privée à partir de ce point. • Mettez à jour votre cluster.

Vous pouvez modifier l'accès aux points de terminaison de votre serveur d'API de cluster à l'aide du AWS Management Console ou AWS CLI.

AWS Management Console

Pour modifier l'accès aux points de terminaison de votre serveur d'API de cluster à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster pour afficher les informations le concernant.
3. Choisissez la page Réseaux, puis Mise à jour.
4. Pour l'accès privé, choisissez si vous souhaitez activer ou désactiver l'accès privé pour le point de terminaison du serveur d'API Kubernetes de votre cluster. Si vous activez l'accès privé, les demandes d'API Kubernetes en provenance du VPC de votre cluster utilisent le point de terminaison d'un VPC privé. Vous devez activer l'accès privé pour désactiver l'accès public.
5. Pour l'accès public, choisissez si vous souhaitez activer ou désactiver l'accès public pour le point de terminaison du serveur d'API Kubernetes de votre cluster. Si vous désactivez l'accès public, le serveur d'API Kubernetes de votre cluster peut recevoir uniquement des demandes provenant du VPC du cluster.
6. (Facultatif) Si vous avez activé l'accès public, vous pouvez spécifier les adresses internet qui peuvent communiquer au point de terminaison public. Sélectionnez Advanced Settings (Paramètres avancés). Entrez un bloc CIDR, tel que *203.0.113.5/32*. Le bloc ne peut pas inclure d'adresses réservées. Vous pouvez entrer des blocs supplémentaires en

sélectionnant Add Source (Ajouter une source). Vous pouvez spécifier un nombre maximal de blocs CIDR. Pour plus d'informations, consultez Service quotas Amazon EKS. Si vous ne spécifiez aucun bloc, le point de terminaison du serveur d'API public reçoit les demandes de toutes les adresses IP (`0.0.0.0/0`). Si vous limitez l'accès à votre point de terminaison public à l'aide de blocs CIDR, il est recommandé d'activer également l'accès au point de terminaison privé afin que les nœuds et les Pods Fargate (si vous les utilisez) puissent communiquer avec le cluster. Sans le point de terminaison privé activé, vos sources CIDR de point d'accès public doivent inclure les sources de sortie de votre VPC. Par exemple, si vous avez un nœud dans un sous-réseau privé qui communique à Internet via une passerelle NAT, vous devez ajouter l'adresse IP sortante de la passerelle NAT dans le cadre d'un bloc CIDR autorisé sur votre point de terminaison public.

7. Choisissez Update (Mettre à jour) pour terminer.

AWS CLI

Pour modifier l'accès au point de terminaison du serveur d'API de votre cluster à l'aide de l' AWS CLI

Effectuez les étapes suivantes en utilisant la AWS CLI version `1.27.160` ou une version ultérieure. Vous pouvez vérifier votre version actuelle avec `aws --version`. Pour installer ou mettre à niveau le AWS CLI, reportez-vous à la section Installation du AWS CLI.

1. Mettez à jour l'accès au point de terminaison du serveur d'API de votre cluster à l'aide de la commande suivante de l' AWS CLI : Remplacez les valeurs par le nom de votre cluster et les valeurs d'accès au point de terminaison souhaitées. Si vous définissez `endpointPublicAccess=true`, vous pouvez (éventuellement) entrer un seul bloc CIDR ou une liste de blocs CIDR séparés par des virgules pour `publicAccessCidrs`. Les blocs ne peuvent pas inclure d'adresses réservées. Si vous spécifiez des blocs CIDR, le point de terminaison du serveur d'API public ne recevra que les demandes des blocs répertoriés. Vous pouvez spécifier un nombre maximal de blocs CIDR. Pour plus d'informations, consultez Service quotas Amazon EKS. Si vous limitez l'accès à votre point de terminaison public à l'aide de blocs CIDR, il est recommandé d'activer également l'accès au point de terminaison privé afin que les nœuds et les Pods Fargate (si vous les utilisez) puissent communiquer avec le cluster. Sans le point de terminaison privé activé, vos sources CIDR de point d'accès public doivent inclure les sources de sortie de votre VPC. Par exemple, si vous avez un nœud dans un sous-réseau privé qui communique à Internet via une passerelle NAT, vous devez ajouter l'adresse IP sortante de la passerelle NAT dans le cadre d'un bloc

CIDR autorisé sur votre point de terminaison public. Si vous ne spécifiez aucun bloc CIDR, le point de terminaison du serveur d'API public reçoit les demandes de toutes les adresses IP (0.0.0.0/0).

> **Note**
>
> La commande suivante active l'accès privé et l'accès public à partir d'une adresse IP unique pour le point de terminaison du serveur API. Remplacez *203.0.113.5/32* par un seul bloc CIDR ou une liste de blocs CIDR séparés par des virgules auxquels vous souhaitez restreindre l'accès réseau.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --resources-vpc-config
 endpointPublicAccess=true,publicAccessCidrs="203.0.113.5/32",endpointPrivateAccess=true
```

L'exemple qui suit illustre un résultat.

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "InProgress",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
```

```
    }
}
```

2. Surveillez le statut de la mise à jour de l'accès à votre point de terminaison avec la commande suivante, en indiquant le nom de votre cluster et l'ID de mise à jour qui a été renvoyé par la commande précédente. Votre mise à jour est terminée lorsqu'elle affiche l'état `Successful`.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000
```

L'exemple qui suit illustre un résultat.

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "Successful",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

Accès à un serveur d'API privé uniquement

Si vous avez désactivé l'accès public pour le point de terminaison du serveur d'API Kubernetes de votre cluster, vous pouvez uniquement accéder au serveur d'API depuis votre VPC ou un réseau connecté. Voici quelques méthodes d'accès possibles au point de terminaison du serveur d'API Kubernetes :

Réseau connecté

Connectez votre réseau au VPC à l'aide d'une passerelle de transit AWS ou d'une autre option de connectivité, puis utilisez un ordinateur dans le réseau connecté. Vous devez vous assurer que le groupe de sécurité de votre plan de contrôle Amazon EKS contient des règles pour autoriser le trafic entrant sur le port 443 depuis votre réseau connecté.

Hôte bastion Amazon EC2

Vous pouvez lancer une instance Amazon EC2 dans un sous-réseau public dans le VPC de votre cluster, puis vous connecter via SSH à cette instance pour exécuter des commandes `kubectl`. Pour plus d'informations, consultez la section Hôtes bastions Linux sur AWS. Vous devez vous assurer que le groupe de sécurité de votre plan de contrôle Amazon EKS contient des règles pour autoriser le trafic entrant sur le port 443 depuis l'hôte bastion. Pour plus d'informations, consultez Considérations et exigences relatives aux groupes de sécurité Amazon EKS.

Lorsque vous configurez `kubectl` pour votre hôte bastion, assurez-vous d'utiliser des informations d'identification AWS déjà mappées à la configuration RBAC de votre cluster, ou ajoutez le principal IAM que votre bastion utilisera à la configuration RBAC avant de supprimer l'accès public au point de terminaison. Pour plus d'informations, consultez the section called "Autoriser l'accès aux API Kubernetes" et Accès non autorisé ou refusé (`kubectl`).

AWS Cloud9 IDE

AWS Cloud9 est un environnement de développement intégré (IDE) basé sur le cloud qui vous permet d'écrire, d'exécuter et de déboguer votre code avec un simple navigateur. Vous pouvez créer un AWS Cloud9 IDE dans le VPC de votre cluster et utiliser cet IDE pour communiquer avec votre cluster. Pour plus d'informations, consultez Création d'un environnement dans AWS Cloud9. Vous devez vous assurer que votre groupe de sécurité de plan de contrôle Amazon EKS contient des règles permettant d'autoriser le trafic entrant sur le port 443 à partir de votre groupe de sécurité IDE. Pour plus d'informations, consultez Considérations et exigences relatives aux groupes de sécurité Amazon EKS.

Lorsque vous configurez `kubectl` votre AWS Cloud9 IDE, veillez à utiliser des AWS informations d'identification déjà mappées à la configuration RBAC de votre cluster, ou ajoutez le principal IAM que votre IDE utilisera à la configuration RBAC avant de supprimer l'accès public du point de terminaison. Pour plus d'informations, consultez Autoriser l'accès aux Kubernetes API et Accès non autorisé ou refusé (`kubectl`).

Activation du chiffrement du secret sur un cluster existant

Si vous activez le chiffrement des secrets, les secrets de Kubernetes sont chiffrés à l'aide du AWS KMS key que vous sélectionnez. La clé KMS doit répondre aux conditions suivantes :

- Symétrique
- Peut chiffrer et déchiffrer des données
- Créée dans la même Région AWS que le cluster
- Si la clé KMS a été créée dans un autre compte, le principal IAM doit avoir accès à la clé KMS.

Pour plus d'informations, consultez la rubrique Autorisation des principaux IAM d'autres comptes à utiliser une clé KMS du Guide du développeur AWS Key Management Service.

> **Warning**
>
> Vous ne pouvez pas désactiver le chiffrement des secrets après l'avoir activé. Cette action est irréversible.

eksctl

Vous pouvez activer le chiffrement de deux manières :

- Ajoutez le chiffrement à votre cluster à l'aide d'une seule commande.

  Pour re-chiffrer automatiquement vos secrets, exécutez la commande suivante.

```
eksctl utils enable-secrets-encryption \
    --cluster my-cluster \
    --key-arn arn:aws:kms:region-code:account:key/key
```

Pour vous désabonner du chiffrement automatique de vos secrets, exécutez la commande suivante.

```
eksctl utils enable-secrets-encryption
    --cluster my-cluster \
    --key-arn arn:aws:kms:region-code:account:key/key \
    --encrypt-existing-secrets=false
```

- Ajoutez le chiffrement à votre cluster à l'aide d'un fichier `kms-cluster.yaml`.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

secretsEncryption:
  keyARN: arn:aws:kms:region-code:account:key/key
```

Pour re-chiffrer automatiquement vos secrets, exécutez la commande suivante.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml
```

Pour vous désabonner du chiffrement automatique de vos secrets, exécutez la commande suivante.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml --encrypt-existing-
secrets=false
```

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le cluster auquel vous souhaitez ajouter le chiffrement KMS.
3. Cliquez sur l'onglet Présentation (cette option est sélectionnée par défaut).

4. Faites défiler la page jusqu'à Secrets encryption (Chiffrement des secrets) et sélectionnez Enable (Activer).
5. Sélectionnez une clé dans la liste déroulant et appuyez sur le bouton Enable (Activer). Si aucune clé n'est répertoriée, vous devez d'abord en créer une. Pour plus d'informations, consultez Création de clés.
6. Appuyez sur le bouton Confirm (Confirmer) pour utiliser la touche choisie.

AWS CLI

1. Associez la configuration Chiffrement des secrets à votre cluster à l'aide de la commande AWS CLI suivante. Remplacez les *example values* par vos propres valeurs.

```
aws eks associate-encryption-config \
    --cluster-name my-cluster \
    --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws:kms:region-code:account:key/key"}}]'
```

L'exemple qui suit illustre un résultat.

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "InProgress",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734,
    "errors": []
  }
}
```

2. Vous pouvez contrôler l'état de votre mise à jour de chiffrement à l'aide de la commande suivante. Utilisez le `cluster name` et le `update ID` spécifiques qui ont été renvoyés dans la sortie précédente. Lorsqu'un état `Successful` s'affiche, la mise à jour est terminée.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id 3141b835-8103-423a-8e68-12c2521ffa4d
```

L'exemple qui suit illustre un résultat.

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "Successful",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734>,
    "errors": []
  }
}
```

3. Pour vérifier que le chiffrement est activé dans votre cluster, exécutez la commande `describe-cluster`. La réponse contient une chaîne `EncryptionConfig`.

```
aws eks describe-cluster --region region-code --name my-cluster
```

Une fois que vous avez activé le chiffrement sur votre cluster, vous devez chiffrer tous les secrets existants avec la nouvelle clé :

> **Note**
>
> Si vous utilisez `eksctl`, l'exécution de la commande suivante n'est nécessaire que si vous vous désabonnez du chiffrement automatique de vos secrets.

```
kubectl get secrets --all-namespaces -o json | kubectl annotate --overwrite -f - kms-encryption-timestamp="time value"
```

Warning

Si vous activez le chiffrement de secrets pour un cluster existant et que la clé KMS que vous utilisez est supprimée, il n'y a pas de chemin vers la récupération pour le cluster. Si vous supprimez la clé KMS, vous mettez définitivement le cluster dans un état dégradé. Pour plus d'informations, consultez Suppression de clés KMS AWS.

Note

Par défaut, la commande `create-key` crée une clé KMS de chiffrement symétrique avec une stratégie de clé qui donne à la racine du compte un accès administrateur sur les actions et les ressources AWS KMS. Si vous souhaitez limiter les autorisations, assurez-vous que les actions `kms:DescribeKey` et `kms:CreateGrant` sont autorisées sur la stratégie pour le principal qui appellera l'API `create-cluster`.
Pour les clusters utilisant le chiffrement d'enveloppe KMS, des autorisations `kms:CreateGrant` sont requises. La condition n'`kms:GrantIsForAWSResource`est pas prise en charge pour l' CreateClusteraction et ne doit pas être utilisée dans les politiques KMS pour contrôler `kms:CreateGrant` les autorisations accordées aux utilisateurs CreateCluster.

Activation de la prise en charge de Windows pour votre cluster Amazon EKS

Avant de déployer des nœuds Windows, veuillez tenir compte des considérations suivantes.

Considérations

- Vous pouvez utiliser le réseau hôte sur les nœuds Windows à l'aide des pods `HostProcess`. Pour plus d'informations, consultez la section Créer un Windows `HostProcess`Pod dans la documentation Kubernetes.
- Les clusters Amazon EKS doivent contenir un ou plusieurs nœuds Linux ou Fargate pour exécuter les Pods du système principal qui ne s'exécutent que sur Linux, comme CoreDNS.

- Les journaux d'événements `kubelet` et `kube-proxy` sont redirigés vers le journal des événements EKS Windows et sont définis sur une limite de 200 Mo.
- Vous ne pouvez pas utiliser Groupes de sécurité pour Pods avec des Pods s'exécutant sur les nœuds Windows.
- Vous ne pouvez pas utiliser la mise en réseau personnalisée avec les nœuds Windows.
- Vous ne pouvez pas utiliser `IPv6` avec les nœuds Windows.
- Les nœuds Windows prennent en charge une interface réseau Elastic par nœud. Par défaut, le nombre de Pods que vous pouvez exécuter par nœud Windows est égal au nombre d'adresses IP disponibles par interface réseau élastique pour le type d'instance du nœud, moins une. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de l'utilisateur Amazon EC2.
- Dans un cluster Amazon EKS, un seul service avec un équilibreur de charge peut prendre en charge jusqu'à 1024 Pods back-end. Chaque Pod a sa propre adresse IP. La limite précédente de 64 Pods n'est plus valable, suite à une mise à jour de Windows Server commençant par la version 17763.2746 du système d'exploitation.
- Les conteneurs Windows ne sont pas pris en charge pour les Pods Amazon EKS sur Fargate.
- Vous ne pouvez pas récupérer les journaux du pod `vpc-resource-controller`. Vous le pouviez auparavant lorsque vous déployiez le contrôleur sur le plan de données.
- Il y a un temps de stabilisation avant qu'une adresse `IPv4` ne soit affectée à un nouveau pod. Cela empêche le trafic de s'écouler vers un pod plus ancien avec la même adresse `IPv4` en raison de règles `kube-proxy` périmées.
- La source du contrôleur est gérée sur GitHub. Pour contribuer ou soumettre des problèmes concernant le contrôleur, consultez le projet sur GitHub.
- Lorsque vous spécifiez un ID AMI personnalisé pour les groupes de nœuds Windows gérés, ajoutez-le `eks:kube-proxy-windows` à votre carte de configuration AWS IAM Authenticator. Pour plus d'informations, consultez Limites et conditions lors de la spécification d'un ID d'AMI.

Prérequis

- Un cluster existant. Le cluster doit exécuter l'une des versions de Kubernetes et de la plateforme répertoriées dans le tableau suivant. Toute version de Kubernetes et de plateforme ultérieure à celles répertoriées est également prise en charge. Si la version de votre cluster ou de votre plateforme est antérieure à l'une des versions suivantes, vous devez activer la prise en charge de

Windows hérité sur le plan de données de votre cluster. Une fois que votre cluster se trouve dans l'une des versions suivantes de Kubernetes et de la plateforme, ou une version ultérieure, vous pouvez supprimer la prise en charge de Windows hérité et activer la prise en charge de Windows sur votre plan de contrôle.

Version de Kubernetes	Version de plateforme
1,30	eks.2
1,29	eks.1
1,28	eks.1
1,27	eks.1
1,26	eks.1
1,25	eks.1
1,24	eks.2

- Votre cluster doit avoir au moins un (nous recommandons au moins deux) nœud Linux ou Pod Fargate pour exécuter CoreDNS. Si vous activez la prise en charge de Windows hérité, vous devez utiliser un nœud Linux (vous ne pouvez pas utiliser un Pod Fargate) pour exécuter CoreDNS.
- Un Rôle IAM de cluster Amazon EKS existant.

Activation de la prise en charge de Windows

Si votre cluster ne se trouve pas dans l'une des versions de Kubernetes et de la plateforme répertoriées dans les prérequis, vous devez plutôt activer la prise en charge de Windows hérité. Pour plus d'informations, consultez Activation de la prise en charge de Windows hérité.

Si vous n'avez jamais activé la prise en charge de Windows sur votre cluster, passez directement à l'étape suivante.

Si vous avez activé la prise en charge de Windows sur un cluster antérieur à une version de Kubernetes ou de plateforme répertoriée dans les prérequis, vous devez d'abord supprimer le `vpc-resource-controller` et le `vpc-admission-webhook` de votre plan de données. Ils sont obsolètes et ne sont plus nécessaires.

Pour activer la prise en charge de Windows sur votre cluster

1. Si vous n'avez pas de nœuds Amazon Linux dans votre cluster et que vous utilisez des groupes de sécurité pour les Pods, passez à l'étape suivante. Sinon, confirmez que la politique gérée `AmazonEKSVPCResourceController` est attachée à votre rôle de cluster. Remplacez *`eksClusterRole`* par le nom de rôle de votre cluster.

   ```
   aws iam list-attached-role-policies --role-name eksClusterRole
   ```

   L'exemple qui suit illustre un résultat.

   ```
   {
       "AttachedPolicies": [
           {
               "PolicyName": "AmazonEKSClusterPolicy",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"
           },
           {
               "PolicyName": "AmazonEKSVPCResourceController",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSVPCResourceController"
           }
       ]
   }
   ```

   Si la politique est attachée, comme c'est le cas dans la sortie précédente, passez à l'étape suivante.

2. Joignez la politique ResourceController gérée par Amazon KSVPC à votre. Rôle IAM de cluster Amazon EKS Remplacez *`eksClusterRole`* par le nom de rôle de votre cluster.

   ```
   aws iam attach-role-policy \
     --role-name eksClusterRole \
     --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
   ```

3. Créez un fichier nommé *`vpc-resource-controller-configmap.yaml`* avec les contenus suivants.

   ```
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: amazon-vpc-cni
   ```

```
  namespace: kube-system
data:
  enable-windows-ipam: "true"
```

4. Appliquer le `ConfigMap` à votre cluster.

```
kubectl apply -f vpc-resource-controller-configmap.yaml
```

5. Vérifiez que votre `ConfigMap` `aws-auth` contient un mappage pour le rôle d'instance du nœud Windows afin d'inclure le groupe d'autorisations RBAC `eks:kube-proxy-windows`. Vous pouvez vérifier en exécutant la commande suivante.

```
kubectl get configmap aws-auth -n kube-system -o yaml
```

L'exemple qui suit illustre un résultat.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      - eks:kube-proxy-windows # This group is required for Windows DNS resolution
 to work
      rolearn: arn:aws:iam::111122223333:role/eksNodeRole
      username: system:node:{{EC2PrivateDNSName}}
[...]
```

Vous devriez voir `eks:kube-proxy-windows` répertorié sous les groupes. Si le groupe n'est pas spécifié, vous devez mettre à jour votre `ConfigMap` ou la créer pour inclure le groupe requis. Pour en savoir plus sur `ConfigMap` `aws-auth`, consultez Appliquer la `ConfigMap` `aws-auth` à votre cluster.

Suppression de la prise en charge de Windows hérité de votre plan de données

Si vous avez activé la prise en charge de Windows sur un cluster antérieur à une version de Kubernetes ou de plateforme répertoriée dans les prérequis, vous devez d'abord supprimer le `vpc-resource-controller` et le `vpc-admission-webhook` de votre plan de données. Ils sont obsolètes et ne sont plus nécessaires, car la fonctionnalité qu'ils fournissaient est maintenant activée sur le plan de contrôle.

1. Désinstallez `vpc-resource-controller` avec la commande suivante. Utilisez cette commande quel que soit l'outil avec lequel vous l'avez installé à l'origine. Remplacez *`region-code`* (seulement l'instance de ce texte après `/manifests/`) par la Région AWS dans laquelle se trouve votre cluster.

   ```
   kubectl delete -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
   ```

2. Désinstallez `vpc-admission-webhook` en suivant les instructions de l'outil avec lequel vous l'avez installé.

   eksctl

   Exécutez les commandes suivantes.

   ```
   kubectl delete deployment -n kube-system vpc-admission-webhook
   kubectl delete service -n kube-system vpc-admission-webhook
   kubectl delete mutatingwebhookconfigurations.admissionregistration.k8s.io vpc-admission-webhook-cfg
   ```

   kubectl on macOS or Windows

   Exécutez la commande suivante. Remplacez *`region-code`* (uniquement l'instance de ce texte après`/manifests/`) par Région AWS celle dans laquelle se trouve votre cluster.

   ```
   kubectl delete -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml
   ```

3. Activez la prise en charge de Windows pour votre cluster sur le plan de contrôle.

Désactivation de la prise en charge de Windows

Pour désactiver la prise en charge de Windows sur votre cluster

1. Si votre cluster contient des nœuds Amazon Linux et que vous utilisez des groupes de sécurité pour les Pods, passez cette étape.

   Supprimez la politique IAM gérée `AmazonVPCResourceController` de votre rôle de cluster. Remplacez *`eksClusterRole`* par le nom du rôle de votre cluster et *`111122223333`* par l'ID de votre compte.

   ```
   aws iam detach-role-policy \
       --role-name eksClusterRole \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
   ```

2. Désactivez Windows IPAM dans le. `amazon-vpc-cni` ConfigMap

   ```
   kubectl patch configmap/amazon-vpc-cni \
                       -n kube-system \
                       --type merge \
                       -p '{"data":{"enable-windows-ipam":"false"}}'
   ```

Déploiement de pods

Lorsque vous déployez des pods dans votre cluster, vous devez spécifier le système d'exploitation qu'ils utilisent si vous exécutez un mélange de types de nœuds.

Pour les Pods Linux, utilisez le texte de sélecteur de nœuds suivant dans vos manifestes.

```
nodeSelector:
        kubernetes.io/os: linux
        kubernetes.io/arch: amd64
```

Pour les Pods Windows, utilisez le texte de sélecteur de nœuds suivant dans vos manifestes.

```
nodeSelector:
        kubernetes.io/os: windows
        kubernetes.io/arch: amd64
```

Vous pouvez déployer un exemple d'application pour voir les sélecteurs de nœuds utilisés.

Activation de la prise en charge de Windows hérité

Si votre cluster se trouve dans l'une des versions de Kubernetes et de la plateforme répertoriées dans les prérequis, nous vous recommandons d'activer plutôt la prise en charge de Windows sur votre plan de contrôle. Pour plus d'informations, consultez Activation de la prise en charge de Windows.

Les étapes suivantes vous aident à activer la prise en charge de Windows hérité pour le plan de données de votre cluster Amazon EKS si la version de votre cluster ou de votre plateforme est antérieure aux versions répertoriées dans les prérequis. Lorsque la version de votre cluster et de votre plateforme est égale ou supérieure à une version répertoriée dans les prérequis, nous vous recommandons de supprimer la prise en charge de Windows hérité et de l'activer pour votre plan de contrôle.

Vous pouvez utiliser `eksctl`, un client Windows ou un client macOS ou Linux pour activer la prise en charge de Windows hérité pour votre cluster.

eksctl

Pour activer la prise en charge de Windows hérité pour votre cluster avec **`eksctl`**

Prérequis

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version à l'aide de la commande suivante.

```
eksctl version
```

Pour plus d'informations sur l'installation ou la mise à niveau de `eksctl`, consultez Installation dans la documentation `eksctl`.

1. Activez la prise en charge de Windows pour votre cluster Amazon EKS avec la commande `eksctl` suivante. Remplacez *`my-cluster`* par le nom de votre cluster. Cette commande déploie le webhook du contrôleur de ressources VPC et du contrôleur d'admission VPC requis sur les clusters Amazon EKS pour exécuter des charges de travail Windows.

   ```
   eksctl utils install-vpc-controllers --cluster my-cluster --approve
   ```

> **Important**
>
> Le point d'ancrage web du contrôleur d'admission VPC est signé avec un certificat qui expire un an après la date d'émission. Pour éviter les temps d'arrêt, assurez-vous de renouveler le certificat avant qu'il n'expire. Pour plus d'informations, consultez Renouvellement du certificat du webhook d'admission VPC.

2. Après avoir activé la prise en charge de Windows, vous pouvez lancer un groupe de nœuds Windows dans votre cluster. Pour plus d'informations, consultez Lancement de nœuds Windows autogérés.

Windows

Pour activer la prise en charge de Windows hérité pour votre cluster avec un client Windows

Dans les étapes suivantes, remplacez le *region-code* par la Région AWS dans laquelle réside votre cluster.

1. Déployez le contrôleur de ressources VPC sur votre cluster.

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. Déployez le webhook du contrôleur d'admission VPC sur votre cluster.

   a. Téléchargez les scripts et les fichiers de déploiement requis.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/Setup-VPCAdmissionWebhook.ps1;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.ps1;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-patch-ca-bundle.ps1;
```

   b. Installez OpenSSL et jq.

   c. Configurez et déployez le webhook d'admission VPC.

```
./Setup-VPCAdmissionWebhook.ps1 -DeploymentTemplate ".\vpc-admission-webhook-deployment.yaml"
```

> **Important**
>
> Le point d'ancrage web du contrôleur d'admission VPC est signé avec un certificat qui expire un an après la date d'émission. Pour éviter les temps d'arrêt, assurez-vous de renouveler le certificat avant qu'il n'expire. Pour plus d'informations, consultez Renouvellement du certificat du webhook d'admission VPC.

3. Déterminez si votre cluster possède la liaison de rôle de cluster requise.

```
kubectl get clusterrolebinding eks:kube-proxy-windows
```

Si une sortie similaire à l'exemple suivant est renvoyée, le cluster a la liaison de rôle nécessaire.

```
NAME                       AGE
eks:kube-proxy-windows     10d
```

Si la sortie inclut `Error from server (NotFound)`, le cluster n'a pas la liaison de rôle de cluster nécessaire. Ajoutez la liaison en créant un fichier nommé *`eks-kube-proxy-windows-crb.yaml`* avec le contenu suivant.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
    eks.amazonaws.com/component: kube-proxy
subjects:
  - kind: Group
    name: "eks:kube-proxy-windows"
roleRef:
  kind: ClusterRole
  name: system:node-proxier
```

```
  apiGroup: rbac.authorization.k8s.io
```

Appliquez la configuration au cluster.

```
kubectl apply -f eks-kube-proxy-windows-crb.yaml
```

4. Après avoir activé la prise en charge de Windows, vous pouvez lancer un groupe de nœuds Windows dans votre cluster. Pour plus d'informations, consultez Lancement de nœuds Windows autogérés.

macOS and Linux

Pour activer la prise en charge de Windows hérité pour votre cluster avec un client macOS ou Linux

Cette procédure nécessite que la bibliothèque `openssl` et le processeur JSON `jq` soient installés sur votre système client.

Dans les étapes suivantes, remplacez le *region-code* par la Région AWS dans laquelle réside votre cluster.

1. Déployez le contrôleur de ressources VPC sur votre cluster.

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. Créez le manifeste du webhook du contrôleur d'admission VPC pour votre cluster.

   a. Téléchargez les scripts et les fichiers de déploiement requis.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.sh
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-patch-ca-bundle.sh
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml
```

   b. Ajoutez des autorisations aux scripts shell afin qu'ils puissent être exécutés.

```
chmod +x webhook-create-signed-cert.sh webhook-patch-ca-bundle.sh
```

c. Créez un secret pour sécuriser la communication.

```
./webhook-create-signed-cert.sh
```

d. Vérifiez le secret.

```
kubectl get secret -n kube-system vpc-admission-webhook-certs
```

e. Configurez le webhook et créez un fichier de déploiement.

```
cat ./vpc-admission-webhook-deployment.yaml | ./webhook-patch-ca-bundle.sh >
 vpc-admission-webhook.yaml
```

3. Déployez le webhook d'admission VPC.

```
kubectl apply -f vpc-admission-webhook.yaml
```

> **Important**
>
> Le point d'ancrage web du contrôleur d'admission VPC est signé avec un certificat qui expire un an après la date d'émission. Pour éviter les temps d'arrêt, assurez-vous de renouveler le certificat avant qu'il n'expire. Pour plus d'informations, consultez Renouvellement du certificat du webhook d'admission VPC.

4. Déterminez si votre cluster possède la liaison de rôle de cluster requise.

```
kubectl get clusterrolebinding eks:kube-proxy-windows
```

Si une sortie similaire à l'exemple suivant est renvoyée, le cluster a la liaison de rôle nécessaire.

```
NAME                     ROLE                                AGE
eks:kube-proxy-windows   ClusterRole/system:node-proxier   19h
```

Si la sortie inclut `Error from server (NotFound)`, le cluster n'a pas la liaison de rôle de cluster nécessaire. Ajoutez la liaison en créant un fichier nommé *`eks-kube-proxy-windows-crb.yaml`* avec le contenu suivant.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
    eks.amazonaws.com/component: kube-proxy
subjects:
  - kind: Group
    name: "eks:kube-proxy-windows"
roleRef:
  kind: ClusterRole
  name: system:node-proxier
  apiGroup: rbac.authorization.k8s.io
```

Appliquez la configuration au cluster.

```
kubectl apply -f eks-kube-proxy-windows-crb.yaml
```

5. Après avoir activé la prise en charge de Windows, vous pouvez lancer un groupe de nœuds Windows dans votre cluster. Pour plus d'informations, consultez Lancement de nœuds Windows autogérés.

Renouvellement du certificat du webhook d'admission VPC

Le certificat utilisé par le webhook d'admission VPC expire un an après la délivrance. Pour éviter les temps d'arrêt, il est important de renouveler le certificat avant qu'il n'expire. Vous pouvez vérifier la date d'expiration de votre certificat actuel à l'aide de la commande suivante.

```
kubectl get secret \
    -n kube-system \
    vpc-admission-webhook-certs -o json | \
    jq -r '.data."cert.pem"' | \
    base64 -decode | \
    openssl x509 \
    -noout \
    -enddate | \
    cut -d= -f2
```

L'exemple qui suit illustre un résultat.

```
May 28 14:23:00 2022 GMT
```

Vous pouvez renouveler le certificat à l'aide de `eksctl` ou d'un ordinateur Windows ou Linux/macOS. Suivez les instructions de l'outil que vous avez utilisé à l'origine pour installer le webhook d'admission VPC. Par exemple, si vous avez installé à l'origine le webhook d'admission VPC en utilisant `eksctl`, vous devez renouveler le certificat à l'aide des instructions figurant sur l'onglet `eksctl`.

eksctl

1. Réinstallez le certificat. Remplacez *`my-cluster`* par le nom de votre cluster.

   ```
   eksctl utils install-vpc-controllers -cluster my-cluster -approve
   ```

2. Vérifiez que vous recevez la sortie suivante.

   ```
   2021/05/28 05:24:59 [INFO] generate received request
   2021/05/28 05:24:59 [INFO] received CSR
   2021/05/28 05:24:59 [INFO] generating key: rsa-2048
   2021/05/28 05:24:59 [INFO] encoded CSR
   ```

3. Redémarrez le déploiement de webhook.

   ```
   kubectl rollout restart deployment -n kube-system vpc-admission-webhook
   ```

4. Si le certificat que vous avez renouvelé a expiré et que vous avez des Pods Windows bloqués dans l'état `Container creating`, vous devez supprimer et redéployer ces Pods.

Windows

1. Obtenez le script pour générer un nouveau certificat.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.ps1;
   ```

2. Préparez le paramètre pour le script.

   ```
   ./webhook-create-signed-cert.ps1 -ServiceName vpc-admission-webhook-svc -SecretName vpc-admission-webhook-certs -Namespace kube-system
   ```

3. Redémarrez le déploiement de webhook.

```
kubectl rollout restart deployment -n kube-system vpc-admission-webhook-deployment
```

4. Si le certificat que vous avez renouvelé a expiré et que vous avez des Pods Windows bloqués dans l'état `Container creating`, vous devez supprimer et redéployer ces Pods.

Linux and macOS

Prérequis

OpenSSL et `jq` doivent être installés sur votre ordinateur.

1. Obtenez le script pour générer un nouveau certificat.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.sh
```

2. Modifiez les autorisations.

```
chmod +x webhook-create-signed-cert.sh
```

3. Exécutez le script.

```
./webhook-create-signed-cert.sh
```

4. Redémarrez le webhook.

```
kubectl rollout restart deployment -n kube-system vpc-admission-webhook-deployment
```

5. Si le certificat que vous avez renouvelé a expiré et que vous avez des Pods Windows bloqués dans l'état `Container creating`, vous devez supprimer et redéployer ces Pods.

Prise en charge d'une densité de Pod plus élevée sur les nœuds Windows

Dans Amazon EKS, chaque `IPv4` se voit attribuer une adresse Pod à partir de votre VPC. De ce fait, le nombre de Pods que vous pouvez déployer sur un nœud est limité par les adresses IP disponibles, même si les ressources sont suffisantes pour exécuter plus de Pods sur le nœud. Étant donné qu'une

seule interface réseau élastique est prise en charge par un nœud Windows, par défaut, le nombre maximal d'adresses IP disponibles sur un nœud Windows est égal à :

```
Number of private IPv4 addresses for each interface on the node - 1
```

Une adresse IP est utilisée comme adresse IP principale de l'interface réseau, elle ne peut donc pas être attribuée à Pods.

Vous pouvez activer une densité de Pod plus élevée sur les nœuds Windows en activant la délégation de préfixes IP. Cette fonctionnalité vous permet d'attribuer un préfixe `/28` `IPv4` à l'interface réseau principale, au lieu d'attribuer des adresses secondaires `IPv4`. L'attribution d'un préfixe IP augmente le nombre maximum d'adresses `IPv4` disponibles sur le nœud à :

```
(Number of private IPv4 addresses assigned to the interface attached to the node - 1) *
 16
```

Avec ce nombre considérablement plus élevé d'adresses IP disponibles, les adresses IP disponibles ne devraient pas limiter votre capacité à augmenter le nombre de Pods sur vos nœuds. Pour plus d'informations, voir Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.

Exigences relatives aux clusters privés

Cette rubrique décrit comment déployer un cluster Amazon EKS déployé sur le AWS Cloud, mais ne disposant pas d'un accès Internet sortant. Si vous avez un cluster local activé AWS Outposts, consultezLancement de nœuds Amazon Linux autogérés sur un Outpost, au lieu de cette rubrique.

Si vous ne connaissez pas la mise en réseau Amazon EKS, consultez la page Démystification de la mise en réseau de cluster pour les composants master Amazon EKS. Si votre cluster n'a pas d'accès Internet sortant, il doit répondre aux exigences suivantes :

- Votre cluster doit extraire des images d'un registre de conteneurs qui se trouve dans votre VPC. Vous pouvez créer un registre de conteneurs Amazon Elastic Container Registry dans votre VPC et y copier des images de conteneurs pour que vos nœuds puissent les extraire. Pour plus d'informations, consultez Copier une image de conteneur d'un référentiel vers un autre référentiel.
- Votre cluster doit avoir un accès privé au point de terminaison activé. Ceci est nécessaire pour que les nœuds s'enregistrent auprès du point de terminaison du cluster. L'accès public au point de terminaison est facultatif. Pour plus d'informations, consultez Contrôle d'accès au point de terminaison du cluster Amazon EKS..

- Les nœuds autogérés Linux et Windows doivent inclure les arguments d'amorçage suivants avant leur lancement. Ces arguments contournent l'introspection Amazon EKS et ne nécessitent pas d'accès à l'API Amazon EKS depuis le VPC.

  1. Déterminez la valeur du point de terminaison de votre cluster à l'aide de la commande suivante. Remplacez *my-cluster* par le nom de votre cluster.

     ```
     aws eks describe-cluster --name my-cluster --query cluster.endpoint --output text
     ```

     L'exemple qui suit illustre un résultat.

     ```
     https://EXAMPLE108C897D9B2F1B21D5EXAMPLE.sk1.region-code.eks.amazonaws.com
     ```

  2. Déterminez la valeur de l'autorité de certification de votre cluster à l'aide de la commande suivante. Remplacez *my-cluster* par le nom de votre cluster.

     ```
     aws eks describe-cluster --name my-cluster --query cluster.certificateAuthority --output text
     ```

     La sortie renvoyée est une longue chaîne.

  3. Remplacez *cluster-endpoint* et *certificate-authority* dans les commandes suivantes par les valeurs renvoyées dans la sortie des commandes précédentes. Pour plus d'informations sur la spécification des arguments d'amorçage lors du lancement de nœuds autogérés, consultez Lancement de nœuds Amazon Linux autogérés et Lancement de nœuds Windows autogérés.

  - Pour les nœuds Linux :

    ```
    --apiserver-endpoint cluster-endpoint --b64-cluster-ca certificate-authority
    ```

    Pour des arguments supplémentaires, consultez le script d'amorçage sur GitHub.

  - Pour les nœuds Windows :

    > **Note**
    >
    > Si vous utilisez un CIDR de service personnalisé, vous devez le spécifier à l'aide du paramètre `-ServiceCIDR`. Dans le cas contraire, la résolution DNS pour les Pods dans le cluster échouera.

```
-APIServerEndpoint cluster-endpoint -Base64ClusterCA certificate-authority
```

Pour des arguments supplémentaires, consultez Paramètres de configuration du script d'amorçage.

- La `ConfigMap aws-auth` de votre cluster doit être créée depuis votre VPC. Pour plus d'informations sur la création et l'ajout d'entrées dans la `ConfigMap aws-auth`, entrez **`eksctl create iamidentitymapping --help`** dans votre terminal. Si la `ConfigMap` n'existe pas sur votre serveur, `eksctl` la créera lorsque vous utiliserez la commande pour ajouter un mappage d'identité.
- Les Pods configurés avec des rôles IAM pour les comptes de service acquièrent des informations d'identification à partir d'un appel d'API AWS Security Token Service (AWS STS). S'il n'existe aucun accès Internet sortant, vous devez créer et utiliser un point de terminaison AWS STS VPC dans votre VPC. La plupart des AWS `v1` SDK utilisent le point de AWS STS terminaison global par défaut (`sts.amazonaws.com`), qui n'utilise pas le point de terminaison AWS STS VPC. Pour utiliser le point de terminaison AWS STS VPC, vous devrez peut-être configurer votre SDK pour utiliser le point de AWS STS terminaison régional (). `sts.region-code.amazonaws.com` Pour plus d'informations, consultez Configuration du AWS Security Token Service point de terminaison pour un compte de service.

- Les sous-réseaux VPC de votre cluster doivent avoir un point de terminaison de l'interface VPC pour toutes les Services AWS auxquelles vos Pods doivent accéder. Pour plus d'informations, consultez Accès à un service AWS à l'aide du point de terminaison d'un VPC d'interface. Certains services et points de terminaison couramment utilisés sont répertoriés dans le tableau suivant. Pour une liste complète des points de terminaison, consultez Services AWS qui s'intègrent à AWS PrivateLink dans le Guide AWS PrivateLink.

Service	Point de terminaison
Amazon EC2	com.amazonaws.*region-code* .ec2
Amazon Elastic Container Registry (pour extraire des images de conteneurs)	com.amazonaws.*region-code* .ecr.api, com.amazonaws.*region-code* .ecr.dkr, and com.amazonaws.*region-code* .s3

Service	Point de terminaison
Équilibreurs de charge Application Load Balancer et Network Load Balancer	com.amazonaws.*region-code* .elasticloadbalancing
AWS X-Ray	com.amazonaws.*region-code* .xray
Amazon CloudWatch Logs	com.amazonaws.*region-code* .logs
AWS Security Token Service (obligatoire lors de l'utilisation de rôles IAM pour les comptes de service)	com.amazonaws.*region-code* .sts

Considérations

- Tous les nœuds autogérés doivent être déployés sur des sous-réseaux qui disposent des points de terminaison de l'interface VPC dont vous avez besoin. Si vous créez un groupe de nœuds gérés, le groupe de sécurité des points de terminaison de l'interface VPC doit autoriser le CIDR pour les sous-réseaux, ou vous devez ajouter le groupe de sécurité des nœuds créé au groupe de sécurité des points de terminaison de l'interface VPC.
- Si vous Pods utilisez des volumes Amazon EFS, avant de les déployerPilote CSI Amazon EFS, le fichier kustomization.yaml du pilote doit être modifié pour configurer les images du conteneur de manière à ce qu'elles soient utilisées de la même manière que Région AWS le cluster Amazon EKS.
- Vous pouvez utiliser le AWS Load Balancer Controllerpour déployer des équilibreurs de charge d' AWS application (ALB) et des équilibreurs de charge réseau sur votre cluster privé. Lors de son déploiement, vous devez utiliser les indicateurs de ligne de commande pour définir `enable-shield`, `enable-waf` et `enable-wafv2` sur false. Certificate discovery (français non disponible) avec les noms d'hôtes des objets Ingress n'est pas prise en charge. Cela est dû au fait que le contrôleur doit atteindre AWS Certificate Manager, qui ne possède pas de point de terminaison d'interface VPC.

  Le contrôleur prend en charge les Network Load Balancers avec des cibles IP, qui sont nécessaires pour une utilisation avec Fargate. Pour plus d'informations, consultez Répartition de la charge des applications sur Amazon EKS et Créer un équilibreur de charge de réseau.

- Cluster Autoscaler est pris en charge. Lors du déploiement de Pods Cluster Autoscaler, assurez-vous que la ligne de commande inclut `--aws-use-static-instance-list=true`. Pour plus d'informations, consultez Utiliser la liste d'instances statiques sur GitHub. Le VPC du nœud de travail doit également inclure le point de terminaison du VPC et le point de terminaison du AWS STS VPC à mise à l'échelle automatique.
- Certains produits logiciels de conteneur utilisent des appels d'API qui accèdent au AWS Marketplace Metering Service pour surveiller l'utilisation. Les clusters privés n'autorisent pas ces appels, vous ne pouvez donc pas utiliser ces types de conteneurs dans les clusters privés.

Versions Kubernetes Amazon EKS

Kubernetes évolue rapidement avec de nouvelles fonctionnalités, des mises à jour de conception et des correctifs de bogues. La communauté publie de nouvelles versions mineures de Kubernetes (telles que la version `1.30`) en moyenne une fois tous les quatre mois. Amazon EKS suit le cycle de publication en amont et d'obsolescence pour les versions mineures. Lorsque de nouvelles versions Kubernetes sont mises à disposition dans Amazon EKS, nous vous recommandons de mettre à jour, de manière proactive, vos clusters et d'utiliser la dernière version disponible.

Une version mineure bénéficie d'un support standard dans Amazon EKS pendant les 14 premiers mois suivant sa sortie. Une fois qu'une version a dépassé la date de fin de support standard, elle entre automatiquement en support étendu pour les 12 mois suivants. Le support étendu vous permet de conserver une version spécifique de Kubernetes plus longtemps moyennant un coût supplémentaire par heure de cluster. Si vous n'avez pas mis à jour votre cluster avant la fin de la période de support étendu, celui-ci est automatiquement mis à niveau vers la plus ancienne version étendue actuellement prise en charge.

Nous vous recommandons de créer votre cluster avec la dernière version disponible de Kubernetes prise en charge par Amazon EKS. Si votre application nécessite une version spécifique de Kubernetes, vous pouvez sélectionner des versions plus anciennes. Vous pouvez créer de nouveaux clusters Amazon EKS sur n'importe quelle version proposée dans le cadre du support standard ou étendu.

Versions disponibles bénéficiant du support standard

Les versions suivantes de Kubernetes bénéficient actuellement du support standard d'Amazon EKS :

- `1.30`

- 1.29
- 1.28
- 1.27
- 1.26

Pour les modifications importantes à prendre en compte pour chaque version bénéficiant du support standard, consultez la rubrique Notes de mise à jour pour les versions bénéficiant du support standard.

Versions disponibles bénéficiant du support étendu

Les versions suivantes de Kubernetes sont actuellement disponibles dans Amazon EKS bénéficiant du support étendu :

- 1.25
- 1.24
- 1.23

Pour les modifications importantes à prendre en compte pour chaque version bénéficiant du support étendu, consultez la rubrique Notes de mise à jour pour les versions bénéficiant d'un soutien étendu.

Les Kubernetes versions suivantes sont actuellement disponibles dans le cadre du support étendu d'Amazon EKS, mais vous ne pouvez pas créer de nouveaux clusters avec ces versions :

- 1.22
- 1.21

Pour plus d'informations sur ces versions, voir Notes de mise à jour pour les versions 1.21 et 1.22

Calendrier de sortie de la version Kubernetes Amazon EKS

Le tableau suivant indique les dates de sortie et de support importantes à prendre en compte pour chaque version de Kubernetes.

> **Note**
>
> Les dates qui comportent uniquement un mois et une année sont approximatives et seront mises à jour lorsqu'une date exacte sera connue.

Version de Kubernetes	Version en amont	Version d'Amazon EKS	Date de fin du support standard	Date de fin du support étendu
1.30	17 avril 2024	23 mai 2024	23 juillet 2025	23 juillet 2026
1.29	13 décembre 2023	23 janvier 2024	23 mars 2025	23 mars 2026
1.28	15 août 2023	26 septembre 2023	26 novembre 2024	26 novembre 2025
1.27	11 avril 2023	24 mai 2023	24 juillet 2024	24 juillet 2025
1.26	9 décembre 2022	11 avril 2023	11 juin 2024	11 juin 2025
1.25	23 août 2022	22 février 2023	1er mai 2024	1er mai 2025
1.24	3 mai 2022	15 novembre 2022	31 janvier 2024	31 janvier 2025
1.23	7 décembre 2021	11 août 2022	11 octobre 2023	11 octobre 2024
1.22	4 août 2021	4 avril 2022	4 juin 2023	1 septembre 2024
1.21	08 avril 2021	19 juillet 2021	16 février 2023	15 juillet 2024

Questions fréquentes concernant les versions d'Amazon EKS

Combien de versions de Kubernetes bénéficient du support standard ?

Conformément à la prise en charge de la communauté Kubernetes des versions de Kubernetes, Amazon EKS s'engage à proposer le support standard d'au moins quatre versions de Kubernetes prêtes pour la production à tout moment. Nous annoncerons la date de fin de support standard

d'une version mineure donnée de Kubernetes au moins 60 jours à l'avance. En raison du processus de qualification et de sortie des nouvelles versions de Kubernetes, la date de fin de prise en charge d'une version de Kubernetes sur Amazon EKS correspondra au jour où le projet Kubernetes cesse de prendre en charge la version en amont, ou au lendemain.

Pendant combien de temps Kubernetes bénéficie-t-il du support standard d'Amazon EKS ?

Une version de Kubernetes bénéficie du support standard pendant 14 mois après avoir été disponible pour la première fois sur Amazon EKS. Cela est d'application même si Kubernetes en amont ne prend plus en charge une version disponible sur Amazon EKS. Nous rétroportons les correctifs de sécurité qui s'appliquent aux versions Kubernetes prises en charge par Amazon EKS.

Suis-je averti lors de la fin du support standard pour une version de Kubernetes sur Amazon EKS ?

Oui. Si l'un des clusters de votre compte exécute la version sur le point de prendre fin, Amazon EKS envoie un avis dans les 12 mois AWS Health Dashboard environ suivant la publication de la Kubernetes version sur Amazon EKS. L'avis indique la date de fin de la prise en charge. Le délai est d'au moins 60 jours à compter de la date de l'avis.

Quelles sont les fonctionnalités Kubernetes prises en charge par Amazon EKS ?

Amazon EKS prend en charge toutes les fonctionnalités de l'API Kubernetes disponibles pour tous (GA). À partir de la version `1.24` de Kubernetes, les nouvelles API bêta ne sont pas activées par défaut dans les clusters. Toutefois, les anciennes API bêta et les nouvelles versions des API bêta existantes continuent d'être activées par défaut. Les fonctions Alpha ne sont pas prises en charge.

Les groupes de nœuds gérés Amazon EKS sont-ils automatiquement mis à jour avec la version du plan de contrôle de cluster ?

Non. Un groupe de nœuds géré crée des instances Amazon EC2 dans votre compte. Ces instances ne sont pas automatiquement mises à niveau lorsque vous ou Amazon EKS mettez à jour votre plan de contrôle. Pour plus d'informations, consultez Mise à jour d'un groupe de nœuds gérés. Nous vous recommandons de conserver la même version Kubernetes sur votre plan de contrôle et vos nœuds.

Les groupes de nœuds autogérés sont-ils automatiquement mis à jour avec la version du plan de contrôle de cluster ?

Non. Un groupe de nœuds autogérés inclut des instances Amazon EC2 dans votre compte. Ces instances ne sont pas mises à niveau automatiquement lorsque la version du plan de contrôle

est mise à jour par vous ou par Amazon EKS en votre nom. Un groupe de nœuds autogérés n'a aucune indication dans la console qu'il doit être mis à jour. Vous pouvez afficher la version `kubelet` installée sur un nœud en sélectionnant le nœud dans la boîte de dialogue Nœuds dans la Présentation de votre cluster pour déterminer quels nœuds doivent être mis à jour. Vous devez mettre à jour les nœuds manuellement. Pour plus d'informations, consultez Mises à jour des nœuds autogérés.

Le projet Kubernetes teste la compatibilité entre le plan de contrôle et les nœuds pour jusqu'à trois versions mineures. Par exemple, les nœuds `1.27` continuent à fonctionner tant qu'ils sont orchestrés par un plan de contrôle `1.30`. Cependant, il n'est pas recommandé d'exécuter un cluster avec des nœuds qui sont plus anciens de trois versions mineures que le plan de contrôle. Pour plus d'informations, consultez Version Kubernetes et politique de prise en charge d'asymétrie de version dans la documentation Kubernetes. Nous vous recommandons de conserver la même version Kubernetes sur votre plan de contrôle et vos nœuds.

Les Pods qui s'exécutent sur Fargate sont-ils automatiquement mis à niveau avec une mise à niveau automatique de la version du plan de contrôle de cluster ?

Non. Nous vous recommandons vivement d'exécuter des Pods Fargate dans le cadre d'un contrôleur de réplication, comme un déploiement Kubernetes. Ensuite, redémarrez tous les Pods Fargate. La nouvelle version du Pod Fargate est déployée avec une version `kubelet` qui est la même que celle de la mise à jour de votre plan de contrôle de cluster. Pour plus d'informations, consultez Déploiements dans la documentation Kubernetes.

> **Important**
>
> Si vous mettez à jour le plan de contrôle, vous devez également mettre à jour les nœuds Fargate vous-même. Pour mettre à jour les nœuds Fargate, supprimez le Pod Fargate représenté par le nœud et redéployez le Pod. Le nouveau Pod est déployé avec une version `kubelet` qui est la même version que celle de votre cluster.

FAQ sur le support étendu Amazon EKS

La terminologie du support standard et du support étendu est nouvelle pour moi. Que signifient ces termes ?

Le support standard pour une version de Kubernetes dans Amazon EKS commence lorsqu'une version de Kubernetes est publiée sur Amazon EKS, et prend fin 14 mois après sa date de sortie.

Le support étendu pour une version de Kubernetes débutera immédiatement après la fin du support standard, et se terminera au bout de 12 mois. Par exemple, le support standard pour les versions `1.23` dans Amazon EKS prend fin le 11 octobre 2023. Le support étendu pour la version `1.23` a débuté le 12 octobre 2023 et se terminera le 11 octobre 2024.

Que dois-je faire pour bénéficier d'un support étendu pour les clusters Amazon EKS ?

Vous n'avez rien à faire pour bénéficier d'un support étendu pour vos clusters Amazon EKS. Le support standard commence lorsqu'une version de Kubernetes est publiée sur Amazon EKS, et prend fin 14 mois après sa date de sortie. Le support étendu pour une version de Kubernetes débutera immédiatement après la fin du support standard, et se terminera au bout de 12 mois. Les clusters qui s'exécutent sur une version de Kubernetes après la fin du support standard bénéficieront automatiquement du support étendu.

Pour quelles versions de Kubernetes puis-je bénéficier d'un support étendu ?

Un support étendu est disponible pour les versions `1.23` et ultérieures de Kubernetes. Vous pouvez exécuter des clusters sur n'importe quelle version jusqu'à 12 mois après la fin du support standard pour cette version. Cela signifie que chaque version sera prise en charge pendant 26 mois dans Amazon EKS (14 mois de support standard plus 12 mois de support étendu).

Et si je ne souhaite pas utiliser le support étendu ?

Si vous ne souhaitez pas bénéficier automatiquement du support étendu, vous pouvez mettre à niveau votre cluster vers une version de Kubernetes compatible avec le support standard d'Amazon EKS. Les clusters qui ne sont pas mis à niveau vers une version de Kubernetes bénéficiant du support standard entreront automatiquement dans la période de support étendu.

Que se passera-t-il à la fin des 12 mois de support étendu ?

Les clusters exécutés sur une version de Kubernetes arrivée au terme de son cycle de vie de 26 mois (14 mois de support standard plus 12 mois de support étendu) seront automatiquement mis à niveau vers la version suivante.

Passé la date de fin de la période de support étendu, vous ne pouvez plus créer de nouveaux clusters Amazon EKS avec la version non prise en charge. Les plans de contrôle existants sont automatiquement mis à jour par Amazon EKS vers la version prise en charge la plus récente, par le biais d'un processus de déploiement progressif après la date de fin de la prise en charge. Après la mise à jour automatique du plan de contrôle, assurez-vous de mettre manuellement à jour les modules complémentaires des clusters et les nœuds Amazon EC2. Pour plus d'informations, consultez Mise à jour de la version Kubernetes de votre cluster Amazon EKS.

Quel est le moment exact auquel mon plan de contrôle est automatiquement mis à jour passé la date de fin de la période de support étendu ?

Amazon EKS ne peut pas fournir de délais spécifiques. Les mises à jour automatiques peuvent intervenir à tout moment après la date de fin de la période de support étendu. Vous ne recevrez aucune notification avant la mise à jour. Nous vous recommandons de mettre à jour, de manière proactive, votre plan de contrôle sans vous fier au processus de mise à jour automatique Amazon EKS. Pour plus d'informations, consultez Mise à jour d'une version Kubernetes de cluster Amazon EKS.

Puis-je laisser mon plan de contrôle sur une version de Kubernetes donnée indéfiniment ?

Non La sécurité du cloud AWS est la priorité absolue. Passé un certain point (généralement 1 an), la communauté Kubernetes arrête de publier des correctifs CVE pour les failles et les vulnérabilités (CVE). La communauté décourage la soumission de CVE pour les versions obsolètes. Cela signifie que les vulnérabilités spécifiques à une ancienne version de Kubernetes pourraient ne pas être signalées. Les clusters sont dès lors exposés sans préavis et aucune option de remédiation en cas de vulnérabilité n'est possible. Par conséquent, Amazon EKS ne laisse pas les plans de contrôle sur une version dont la période de support étendu est terminée.

Y a-t-il des frais supplémentaires pour bénéficier du support étendu ?

Oui, des frais supplémentaires sont facturés pour les clusters Amazon EKS exécutés dans le cadre du support étendu. Pour plus de détails sur les tarifs, consultez le support étendu d'Amazon EKS pour la tarification des Kubernetes versions sur le AWS blog.

Qu'est-ce qui est inclus dans le support étendu ?

Les clusters Amazon EKS bénéficiant du support étendu reçoivent des correctifs de sécurité en continu pour le plan de contrôle Kubernetes. En outre, Amazon EKS publiera des correctifs pour la CNI Amazon VPC, `kube-proxy`, ainsi que les modules complémentaires CoreDNS pour les versions bénéficiant du support étendu. Amazon EKS publiera également des correctifs pour les AMI optimisées Amazon EKS AWS publiées pour Amazon Linux et WindowsBottlerocket, ainsi que pour les nœuds Amazon EKS Fargate pour ces versions. Tous les clusters concernés par le Support étendu continueront d'avoir accès au support technique auprès de AWS.

> **Note**
>
> Support étendu pour les Windows AMI optimisées Amazon EKS publiées par AWS n'est pas disponible pour les Kubernetes versions `1.23` mais est disponible pour les Kubernetes versions `1.24` et supérieures.

Existe-t-il des limites aux correctifs pour les composants non-Kubernetes dans le cadre du support étendu ?

Bien que le support étendu couvre tous les composants Kubernetes spécifiques de AWS, il ne fournira à tout moment un support que pour les AMI optimisées Amazon EKS AWS publiées pour Amazon Linux et Windows. Bottlerocket Cela signifie que vous aurez potentiellement de nouveaux composants (système d'exploitation ou noyau, par exemple) sur votre AMI optimisée pour Amazon EKS lorsque vous utiliserez le support étendu. Par exemple, une fois qu'Amazon Linux 2 aura atteint la fin de son cycle de vie en 2025, les AMI Amazon Linux optimisées pour Amazon EKS seront créées à l'aide d'un système d'exploitation Amazon Linux plus récent. Amazon EKS annoncera et documentera les anomalies importantes du cycle de vie du support telles que celle-ci pour chaque version de Kubernetes.

Puis-je créer de nouveaux clusters à l'aide d'une version bénéficiant d'un support étendu ?

Oui, à l'exclusion de `1.22` et`1.21`. Par exemple, vous pouvez créer un `1.23` cluster, mais pas un `1.22` cluster.

Notes de mise à jour pour les versions bénéficiant du support standard

Cette rubrique présente les modifications importantes à prendre en compte pour chaque version de Kubernetes bénéficiant du support standard. Lors de la mise à niveau, examinez attentivement les modifications apportées entre l'ancienne et la nouvelle version de votre cluster.

> **Note**
>
> Pour les clusters `1.24` et versions ultérieures, les AMI Amazon EKS officiellement publiées incluent `containerd` comme seul environnement d'exécution. Les versions de Kubernetes antérieures à la version `1.24` utilisent Docker comme environnement d'exécution par défaut. Ces versions disposent d'une option d'indicateur d'amorçage que vous pouvez utiliser pour tester vos charges de travail sur n'importe quel cluster pris en charge avec `containerd`. Pour plus d'informations, consultez Amazon EKS a mis fin à la prise en charge de `Dockershim`.

Kubernetes1,30

Kubernetes `1.30` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.30`, consultez l'annonce de la version officielle.

⚠ Important

- À partir de la version Amazon EKS `1.30` ou d'une version plus récente, tous les groupes de nœuds gérés nouvellement créés utiliseront automatiquement par défaut Amazon Linux 2023 (AL2023) comme système d'exploitation des nœuds. Auparavant, les nouveaux groupes de nœuds utilisaient par défaut Amazon Linux 2 (AL2). Vous pouvez continuer à utiliser AL2 en le choisissant comme type d'AMI lors de la création d'un nouveau groupe de nœuds.
  - Pour plus d'informations sur Amazon Linux, consultez la section Comparaison entre AL2 et AL2023 dans le guide de l'utilisateur Amazon Linux.
  - Pour plus d'informations sur la spécification du système d'exploitation d'un groupe de nœuds gérés, consultez. Création d'un groupe de nœuds gérés

- Avec Amazon EKS`1.30`, l'`topology.k8s.aws/zone-id`étiquette est ajoutée aux nœuds de travail. Vous pouvez utiliser les ID de zone de disponibilité (ID AZ) pour déterminer l'emplacement des ressources d'un compte par rapport aux ressources d'un autre compte. Pour plus d'informations, consultez la section ID de zone de disponibilité de vos AWS ressources dans le guide de AWS RAM l'utilisateur.
- À partir de `1.30` là, Amazon EKS n'inclut plus l'`default`annotation sur la `gp2` `StorageClass` ressource appliquée aux clusters nouvellement créés. Cela n'a aucun impact si vous référencez cette classe de stockage par son nom. Vous devez prendre des mesures si vous comptez sur une valeur par défaut `StorageClass` dans le cluster. Vous devez les référencer `StorageClass` par leur nom`gp2`. Vous pouvez également déployer la classe de stockage par défaut recommandée par Amazon EBS en définissant le `defaultStorageClass.enabled` paramètre sur true lors de l'installation `v1.31.0` ou ultérieurement du`aws-ebs-csi-driver add-on`.
- La politique IAM minimale requise pour le rôle IAM du cluster Amazon EKS a changé. L'action `ec2:DescribeAvailabilityZones` est requise. Pour plus d'informations, consultez Rôle IAM de cluster Amazon EKS.

Pour obtenir le journal complet des modifications de Kubernetes `1.30`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.30.md.

Kubernetes1,29

Kubernetes `1.29` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.29`, consultez l'annonce de la version officielle.

> **Important**
>
> - Les versions d'`flowcontrol.apiserver.k8s.io/v1beta2`API obsolètes de `FlowSchema` et dans lesquelles `PriorityLevelConfiguration` elles ne sont plus diffusées. Kubernetes `v1.29` Si vous avez des manifestes ou un logiciel client qui utilise le groupe d'API bêta obsolète, vous devez les modifier avant de procéder à la mise à niveau vers. `v1.29`

- Le `.status.kubeProxyVersion` champ pour les objets de nœud est désormais obsolète, et le Kubernetes projet propose de supprimer ce champ dans une future version. Le champ obsolète n'est pas précis et a toujours été géré par `kubelet` - qui ne connaît pas réellement la `kube-proxy` version, ni même si elle `kube-proxy` est en cours d'exécution. Si vous avez utilisé ce champ dans un logiciel client, arrêtez. Les informations ne sont pas fiables et le champ est désormais obsolète.
- Kubernetes`1.29`Afin de réduire la surface d'attaque potentielle, la `LegacyServiceAccountTokenCleanUp` fonctionnalité considère les anciens jetons secrets générés automatiquement comme non valides s'ils n'ont pas été utilisés pendant une longue période (1 an par défaut) et les supprime automatiquement si leur utilisation n'est pas tentée pendant une longue période après avoir été marqués comme non valides (1 an supplémentaire par défaut). Pour identifier ces jetons, vous pouvez exécuter :

```
kubectl get cm kube-apiserver-legacy-service-account-token-tracking -nkube-system
```

Pour obtenir le journal complet des modifications de Kubernetes `1.29`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.29.md#changelog-since-v1280.

Kubernetes1,28

Kubernetes `1.28` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.28`, consultez l'annonce de la version officielle.

- Kubernetes `v1.28` a étendu la prise en charge du décalage entre les composants du nœud principal et du plan de contrôle d'une version mineure (de `n-2` à `n-3`) afin que les composants du nœud (`kubelet` et `kube-proxy`) de la version mineure prise en charge la plus ancienne puissent fonctionner avec les composants du plan de contrôle (`kube-apiserver`, `kube-scheduler`, `kube-controller-manager` et `cloud-controller-manager`) de la version mineure prise en charge la plus récente.
- Les métriques `force_delete_pods_total` et `force_delete_pod_errors_total` du `Pod GC Controller` ont été améliorées pour tenir compte de toutes les suppressions de pods forcées. Une raison est ajoutée à la métrique pour indiquer si le pod est supprimé de force parce qu'il est fermé, orphelin, altéré ou parce qu'il s'arrête et n'est out-of-service pas planifié.
- Le contrôleur `PersistentVolume (PV)` a été modifié pour attribuer automatiquement une `StorageClass` par défaut à toute `PersistentVolumeClaim` non associée dont le paramètre `storageClassName` n'est pas défini. En outre, le mécanisme de validation des admissions de `PersistentVolumeClaim` du serveur d'API a été ajusté pour permettre de remplacer un état non défini par un nom de `StorageClass` .

Pour obtenir le journal complet des modifications de Kubernetes `1.28`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.28.md#changelog-since-v1270.

Kubernetes1,27

Kubernetes `1.27` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.27`, consultez l'annonce de la version officielle.

> **Important**
>
> - La prise en charge des annotations alpha `seccomp.security.alpha.kubernetes.io/pod` et `container.seccomp.security.alpha.kubernetes.io`des annotations `seccomp` a été supprimée. Les annotations alpha `seccomp` sont devenues obsolètes en `1.19`, avec leur suppression de `1.27`, les champs `seccomp` ne seront plus remplis automatiquement pour les `Pods` avec les annotations `seccomp`. Utilisez plutôt le champ `securityContext.seccompProfile` réservé aux conteneurs ou `Pods` pour configurer les profils `seccomp`. Pour vous assurer que vous utilisez la base d'annotations alpha `seccomp` obsolètes dans votre cluster, exécutez la commande suivante :

```
kubectl get pods --all-namespaces -o json | grep
 -E 'seccomp.security.alpha.kubernetes.io/pod|
container.seccomp.security.alpha.kubernetes.io'
```

- L'argument de ligne de commande `--container-runtime` pour `kubelet` a été supprimé. L'environnement d'exécution du conteneur par défaut pour Amazon EKS a été `containerd` défini depuis`1.24`, ce qui élimine le besoin de spécifier le temps d'exécution du conteneur. À partir de `1.27`, Amazon EKS ignorera l'argument `--container-runtime` transmis à tous les scripts d'amorçage. Il est important de ne pas transmettre cet argument à `--kubelet-extra-args` afin d'éviter les erreurs lors du processus d'amorçage du nœud. Vous devez supprimer l'argument `--container-runtime` de tous vos flux de travail de création de nœuds et de vos scripts de génération.

- Le `kubelet` dans Kubernetes `1.27` a augmenté la valeur par défaut `kubeAPIQPS` à `50` et `kubeAPIBurst` à `100`. Ces améliorations permettent à `kubelet` de gérer un plus grand volume de requêtes d'API, améliorant ainsi les temps de réponse et les performances. Lorsque les demandes de `Pods` augmentent, en raison des exigences de mise à l'échelle, les valeurs par défaut révisées garantissent la capacité de `kubelet` à gérer efficacement l'augmentation de la charge de travail. Par conséquent, les lancements Pod sont plus rapides et les opérations de cluster sont plus efficaces.
- Vous pouvez utiliser une topologie Pod plus fine pour diffuser des politiques telles que `minDomain`. Ce paramètre vous permet de spécifier le nombre minimum de domaines sur lesquels vos `Pods` devez être répartis. `nodeAffinityPolicy` et `nodeTaintPolicy` permettent un niveau de granularité supplémentaire dans la gestion de la distribution Pod. Ceci est conforme aux affinités des nœuds, aux rejets et au champ `matchLabelKeys` dans le `topologySpreadConstraints` de votre spécification `Pod's`. Cela permet de sélectionner des `Pods` pour étaler les calculs après une mise à niveau progressive.
- Kubernetes`1.27` promu en version bêta un nouveau mécanisme de politique `StatefulSets` contrôlant la durée de vie de leur `PersistentVolumeClaims` (`PVCs`). La nouvelle politique de rétention PVC vous permet de spécifier si les données `PVCs` générées à partir du modèle de spécification `StatefulSet` seront automatiquement supprimées ou retenues lors de la suppression du `StatefulSet` ou de la réduction des répliques dans le `StatefulSet`.
- L'option `goaway-chance` dans le serveur d'API Kubernetes permet d'éviter que les connexions client `HTTP/2` soient bloquées sur une seule instance de serveur d'API, en fermant une connexion de manière aléatoire. Lorsque la connexion est fermée, le client essaiera de se reconnecter et

atterrira probablement sur un autre serveur d'API en raison de l'équilibrage de charge. La version `1.27` de Amazon EKS a activé l'indicateur `goaway-chance`. Si votre charge de travail exécutée sur le cluster Amazon EKS utilise un client qui n'est pas compatible avec `HTTP GOAWAY`, nous vous recommandons de mettre à jour votre client pour qu'il gère `GOAWAY` en se reconnectant à la fin de la connexion.

Pour obtenir le journal complet des modifications de Kubernetes `1.27`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.27.md#changelog-since-v1260.

Kubernetes1,26

Kubernetes `1.26` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.26`, consultez l'annonce de la version officielle.

> **Important**
>
> Kubernetes `1.26` ne prend plus en charge CRI `v1alpha2`. Il en résulte que le `kubelet` n'enregistre plus le nœud si l'environnement d'exécution du conteneur ne prend pas en charge CRI `v1`. Cela signifie également que Kubernetes `1.26` ne prend pas en charge la version mineure de containerd `1.5` et antérieures. Si vous utilisez containerd, vous devez effectuer une mise à niveau vers la version containerd`1.6.0` ou une version ultérieure avant de procéder à la mise à niveau de tout nœud vers Kubernetes `1.26`. Vous devez également mettre à niveau tous les autres environnements d'exécution de conteneur qui ne prennent en charge que `v1alpha2`. Pour plus d'informations, reportez-vous au fournisseur du moteur d'exécution du conteneur. Par défaut, AMI Amazon Linux et Bottlerocket incluent la version de containerd `1.6.6`.

- Avant de procéder à la mise à niveau vers Kubernetes `1.26`, mettez à niveau votre Amazon VPC CNI plugin for Kubernetes vers la version `1.12` ou ultérieure. Si vous n'effectuez pas la mise à niveau vers la version `1.12` ou ultérieure du Amazon VPC CNI plugin for Kubernetes, le Amazon VPC CNI plugin for Kubernetes plantera. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.
- L'option `goaway-chance` dans le serveur d'API Kubernetes permet d'éviter que les connexions client `HTTP/2` soient bloquées sur une seule instance de serveur d'API, en fermant une connexion de manière aléatoire. Lorsque la connexion est fermée, le client essaiera de se reconnecter et

atterrira probablement sur un autre serveur d'API en raison de l'équilibrage de charge. La version `1.26` de Amazon EKS a activé l'indicateur `goaway-chance`. Si votre charge de travail exécutée sur le cluster Amazon EKS utilise un client qui n'est pas compatible avec `HTTP GOAWAY`, nous vous recommandons de mettre à jour votre client pour qu'il gère `GOAWAY` en se reconnectant à la fin de la connexion.

Pour obtenir le journal complet des modifications de Kubernetes `1.26`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.26.md#changelog-since-v1250.

Notes de mise à jour pour les versions bénéficiant d'un soutien étendu

Cette rubrique présente les modifications importantes à prendre en compte pour chaque version de Kubernetes bénéficiant d'un support étendu. Lors de la mise à niveau, examinez attentivement les modifications apportées entre l'ancienne et la nouvelle version de votre cluster.

Kubernetes1,25

Kubernetes `1.25` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.25`, consultez l'annonce de la version officielle.

Important

- À partir de Kubernetes version `1.25`, vous ne pourrez plus utiliser les instances `P2` Amazon EC2 avec les instances AMI Amazon Linux accélérées et optimisées pour Amazon EKS prêtes à l'emploi. Ces AMI pour Kubernetes `1.25` ou les versions ultérieures prendront en charge les pilotes de la série `NVIDIA 525` ou ultérieurs, qui sont incompatibles avec les instances `P2`. Toutefois, les pilotes de la série `NVIDIA 525` ou ultérieurs sont compatibles avec les instances `P3`, `P4` et `P5`, ce qui vous permet d'utiliser ces instances avec les AMI pour Kubernetes `1.25` ou une version ultérieure. Avant de mettre à niveau vos clusters Amazon EKS vers la version `1.25`, migrez toutes les instances `P2` vers les instances `P3`, `P4` et `P5`. Vous devez également mettre à jour vos applications de manière proactive pour qu'elles fonctionnent avec les pilotes de la série `NVIDIA 525` ou d'une série ultérieure. Nous prévoyons de rétroporter les nouvelles `NVIDIA 525` séries ou les pilotes ultérieurs vers des Kubernetes versions ultérieures `1.23` et `1.24` ce, fin janvier 2024.

- `PodSecurityPolicy` (PSP) est supprimé dans Kubernetes `1.25`. Les PSPs sont remplacées par la fonctionnalité PSA (Pod Security Admission) et les normes de sécurité des pods (PSS). PSA est un contrôleur d'admission intégré qui met en œuvre les contrôles de sécurité décrits dans la PSS. PSA et PSS passent au statut stable dansKubernetes `1.25` et sont activés dans Amazon EKS par défaut. Si vous PSPs en avez un dans votre cluster, assurez-vous de migrer PSP vers la solution intégrée Kubernetes PSS ou vers une policy-as-code solution avant de passer à la version de votre cluster`1.25`. Dans le cas contraire, vos charges de travail sont susceptibles d'être interrompues. Pour plus d'informations, consultez le FAQ sur la suppression de la politique de sécurité des pods (PSP).
- La version `1.25` de Kubernetes contient des modifications qui modifient le comportement d'une fonctionnalité existante appelée API Priority and Fairness (APF). APF sert à protéger le serveur d'API d'une surcharge potentielle pendant les périodes d'augmentation du volume de demandes. Pour ce faire, elle impose des restrictions sur le nombre de demandes simultanées pouvant être traitées à un moment donné. Ceci est réalisé en appliquant des niveaux de priorité et des limites distincts aux demandes provenant de différentes charges de travail ou utilisateurs. Cette approche assure un traitement privilégié aux applications critiques ou aux demandes à priorité élevée, tout en évitant que les demandes à faible priorité ne surchargent le serveur d'API. Pour plus d'informations, voir API Priority and Fairness dans la documentation Kubernetes ou API Priority and Fairness dans le guide des meilleures pratiques EKS.

  Ces mises à jour ont été introduites dans PR #10352 et PR #118601. Auparavant, APF traitait tous les types de demandes de manière homogène, chaque demande consommant une seule unité de la limite de demandes simultanées. La modification du comportement d'APF accorde des niveaux de concurrence plus élevés aux demandes `LIST` en raison de la charge particulièrement lourde qu'elles imposent au serveur d'API. Le serveur d'API estime le nombre d'objets qui seront renvoyés par une demande `LIST`. Il attribue une unité de concurrence proportionnelle au nombre d'objets renvoyés.

  Suite à la mise à niveau vers la version `1.25` ou supérieure d'Amazon EKS, ce comportement mis à jour pourrait entraîner que les charges de travail avec des demandes `LIST` lourdes (qui fonctionnaient auparavant sans problème) rencontrent des limitations de débit. Cela serait signalé par un code de réponse HTTP 429. Pour éviter toute perturbation potentielle des charges de travail en raison de la limitation du débit des demandes `LIST`, nous vous encourageons vivement à restructurer vos charges de travail afin de réduire le débit de ces demandes. Une autre manière de remédier à cette situation

est d'adapter les paramètres APF pour allouer davantage de capacité aux demandes essentielles tout en réduisant la capacité allouée aux demandes non essentielles. Pour plus d'informations sur ces techniques d'atténuation, voir Prévention des demandes abandonnées dans le guide des meilleures pratiques EKS.

- Amazon EKS`1.25` inclut des améliorations de l'authentification des clusters qui contiennent des bibliothèques YAML mises à jour. Si une valeur YAML de la `ConfigMap` `aws-auth` trouvée dans l'espace de noms `kube-system` commence par une macro, où le premier caractère est une accolade, vous devez ajouter des guillemets (`" "`) avant et après les accolades (`{ }`). Cela est nécessaire pour garantir que la version `aws-iam-authenticator` de `v0.6.3` analyse correctement la `ConfigMap` `aws-auth` dans Amazon EKS `1.25`.
- La version bêta de l'API (`discovery.k8s.io/v1beta1`) de `EndpointSlice` est devenue obsolète dans Kubernetes `1.21` et n'est plus disponible depuis Kubernetes `1.25`. Cette API a été mise à jour vers `discovery.k8s.io/v1`. Pour plus d'informations, consultez la section `EndpointSlice`dans la documentation Kubernetes. Le AWS Load Balancer Controller `v2.4.6` et les versions antérieures utilisaient le point de terminaison `v1beta1` pour communiquer avec `EndpointSlices`. Si vous utilisez la configuration `EndpointSlices` pour AWS Load Balancer Controller, vous devez effectuer la mise à niveau vers AWS Load Balancer Controller `v2.4.7` avant de mettre à niveau votre cluster Amazon EKS vers `1.25`. Si vous effectuez une mise à niveau vers la version `1.25` alors que vous utilisez la configuration `EndpointSlices` du AWS Load Balancer Controller, le contrôleur se bloquera, ce qui entraînera des interruptions de vos charges de travail. Pour mettre à niveau le contrôleur, consultez la rubrique Qu'est-ce que AWS Load Balancer Controller ?.

- `SeccompDefault` passe en version bêta dans Kubernetes `1.25`. En définissant l'indicateur `--seccomp-default` lors de la configuration de `kubelet`, l'environnement d'exécution de conteneur utilise son profil `seccomp` `RuntimeDefault` au lieu du mode non confiné (`seccomp` `disabled`). Les profils par défaut fournissent un ensemble solide de paramètres de sécurité par défaut, tout en préservant l'intégrité de la charge de travail. Bien que cet indicateur soit disponible, Amazon EKS ne l'active pas par défaut, de sorte que le comportement d'Amazon EKS reste effectivement inchangé. Si vous le souhaitez, vous pouvez l'activer sur vos nœuds. Pour plus d'informations, consultez le didacticiel Restriction des appels système d'un conteneur avec seccomp de la documentation Kubernetes.

- L'interface CRI (Container Runtime Interface) pour Docker (ou `Dockershim`) n'est plus prise en charge à partir de Kubernetes `1.24`. Le seul environnement d'exécution de conteneur dans les AMIs officielles Amazon EKS pour les clustersKubernetes `1.24` et les versions ultérieures est `containerd`. Avant de passer à Amazon EKS `1.24` ou à une version ultérieure, supprimez toute référence aux indicateurs de script d'amorçage qui ne sont plus pris en charge. Pour plus d'informations, consultez Amazon EKS a mis fin à la prise en charge de `Dockershim`.
- La prise en charge des requêtes génériques est devenue obsolète dans CoreDNS `1.8.7` et a été supprimée dans CoreDNS `1.9`. Cela a été fait par mesure de sécurité. Les requêtes génériques ne fonctionnent plus et renvoient `NXDOMAIN` au lieu d'une adresse IP.
- L'option `goaway-chance` dans le serveur d'API Kubernetes permet d'éviter que les connexions client `HTTP/2` soient bloquées sur une seule instance de serveur d'API, en fermant une connexion de manière aléatoire. Lorsque la connexion est fermée, le client essaiera de se reconnecter et atterrira probablement sur un autre serveur d'API en raison de l'équilibrage de charge. La version `1.25` de Amazon EKS a activé l'indicateur `goaway-chance`. Si votre charge de travail exécutée sur le cluster Amazon EKS utilise un client qui n'est pas compatible avec `HTTP GOAWAY`, nous vous recommandons de mettre à jour votre client pour qu'il gère `GOAWAY` en se reconnectant à la fin de la connexion.

Pour obtenir le journal complet des modifications de Kubernetes `1.25`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.25.md#changelog-since-v1240.

Kubernetes1,24

Kubernetes `1.24` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.24`, consultez l'annonce de la version officielle.

> **Important**
>
> - À partir de la version `1.24` de Kubernetes, les nouvelles API bêta ne sont pas activées par défaut dans les clusters. Par défaut, les API bêta existantes et les nouvelles versions de celles-ci continuent d'être activées. Amazon EKS adopte le même comportement que l'instance de Kubernetes `1.24` située en amont. Les portails de fonctionnalités qui contrôlent les nouvelles fonctionnalités pour les opérations d'API nouvelles et existantes sont activés par défaut. Ceci est conforme à l'instance de Kubernetes située en amont. Pour plus d'informations, voir KEP-3136 : Les API bêta sont désactivées par défaut. GitHub

- L'interface CRI (Container Runtime Interface) pour Docker (ou Dockershim) n'est plus prise en charge à partir de la version `1.24` de Kubernetes. Les AMI officielles d'Amazon EKS utilisent containerd comme seul environnement d'exécution. Avant de passer à Amazon EKS `1.24` ou version ultérieure, vous devez supprimer toute référence aux indicateurs de script d'amorçage qui ne sont plus pris en charge. Vous devez également vous assurer que le transfert IP est activé pour vos nœuds de travail. Pour plus d'informations, consultez Amazon EKS a mis fin à la prise en charge de `Dockershim`.
- Si vous avez déjà configuré Fluentd pour Container Insights, vous devez effectuer la migration Fluentd vers Fluent Bit avant de mettre à jour votre cluster. Les analyseurs Fluentd sont configurés pour analyser uniquement les messages du journal au format JSON. Contrairement à `dockerd`, l'exécution du conteneur `containerd` a des messages de journal qui ne sont pas au format JSON. Si vous ne migrez pas vers Fluent Bit, certains des analyseurs Fluentd's configurés généreront un grand nombre d'erreurs à l'intérieur du conteneur Fluentd. Pour plus d'informations sur la migration, voir Configurer en Fluent Bit tant que DaemonSet pour envoyer des CloudWatch journaux vers Logs.
- Dans Kubernetes `1.23` et versions antérieures, les certificats de service `kubelet` dotés d'une adresse IP et d'un SAN (Subject Alternative Name) invérifiables étaient automatiquement émis avec un SAN invérifiable. Ces SAN invérifiables sont omis du certificat fourni. Dans les versions `1.24` et ultérieures des clusters, les certificats de service `kubelet` ne sont pas émis si un SAN est invérifiable. Cela entrave le fonctionnement des commandes `kubectl` exec et `kubectl` logs. Pour plus d'informations, consultez Considérations relatives à la signature des certificats avant la mise à niveau de votre cluster vers Kubernetes 1.24.
- Lorsque vous mettez à niveau un cluster Amazon EKS `1.23` qui utilise Fluent Bit, vous devez vous assurer qu'il est exécuté avec la version `k8s/1.3.12` ou une version ultérieure. Vous pouvez le faire en réappliquant le dernier fichier YAML Fluent Bit applicable à partir de GitHub. Pour plus d'informations, consultez la section Configuration Fluent Bit dans le guide de CloudWatch l'utilisateur Amazon.

- Vous pouvez avoir recours à des indicateurs prenant en compte la topologie (Topology Aware Hints) pour spécifier que vous préférez maintenir le trafic dans une zone lorsque les composants master du cluster sont déployés dans plusieurs zones de disponibilité. Le routage du trafic au sein d'une zone peut contribuer à réduire les coûts et à améliorer les performances du réseau. Par défaut, les indicateurs prenant en compte la topologie sont activés dans Amazon EKS `1.24`. Pour

plus d'informations, consultez Indicateurs prenant en compte la topologie dans la documentation Kubernetes.

- Il est prévu que la PSP (`PodSecurityPolicy`) soit supprimée de Kubernetes `1.25`. Les PSPs sont remplacées par la fonctionnalité Pod Security Admission (PSA). PSA est un contrôleur d'admission intégré qui utilise les contrôles de sécurité décrits dans les normes de sécurité des pods (Pod Security Standards - PSS). PSA et PSS sont des fonctionnalités bêta activées par défaut dans Amazon EKS. Pour remédier à la suppression de la PSP dans la version `1.25`, nous vous recommandons d'implémenter PSS dans Amazon EKS. Pour plus d'informations, consultez la section Implémentation des normes de sécurité des pods dans Amazon EKS sur le blog AWS .
- Le `client.authentication.k8s.io/v1alpha1` ExecCredential est retiré dans Kubernetes`1.24`. L' ExecCredential API était généralement disponible en Kubernetes`1.22`. Si vous utilisez un plug-in d'informations d'identification client-go qui repose sur l'API `v1alpha1`, contactez le distributeur de votre plug-in pour savoir comment migrer vers l'API `v1`.
- Pour Kubernetes `1.24`, nous avons ajouté une fonctionnalité au projet Cluster Autoscaler situé en amont afin de simplifier le dimensionnement des groupes de nœuds gérés par Amazon EKS vers et depuis zéro nœud. Auparavant, pour que l'outil Cluster Autoscaler puisse comprendre les ressources, les étiquettes et les rejets d'un groupe de nœuds géré dont la taille était réduite à zéro nœud, vous deviez baliser le groupe Amazon EC2 Auto Scaling sous-jacent avec les détails des nœuds dont il était responsable. Désormais, lorsqu'aucun nœud n'est en cours d'exécution dans le groupe de nœuds géré, Cluster Autoscaler appelle l'opération d'API `DescribeNodegroup` d'Amazon EKS. Cette opération d'API fournit les informations dont Cluster Autoscaler a besoin en termes de ressources, d'étiquettes et de rejets du groupe de nœuds géré. Pour utiliser cette fonctionnalité, vous devez ajouter l'autorisation `eks:DescribeNodegroup` à la politique IAM du compte de service Cluster Autoscaler. Lorsque la valeur d'une balise Cluster Autoscaler figurant sur le groupe Auto Scaling qui alimente un groupe de nœuds géré par Amazon EKS entre en conflit avec le groupe de nœuds lui-même, Cluster Autoscaler préfère la valeur de la balise du groupe Auto Scaling. Cela vous permet de remplacer les valeurs selon vos besoins. Pour plus d'informations, consultez Autoscaling.
- Si vous avez l'intention d'utiliser Inferentia ou d'utiliser des types d'Trainiuminstance avec Amazon EKS`1.24`, vous devez effectuer une mise à niveau vers le plug-in pour AWS Neuron appareil version 1.9.3.0 ou ultérieure. Pour plus d'informations, consultez la version Neuron K8 [1.9.3.0] dans la documentation. AWS Neuron
- `IPv6` est activée par défaut dans `Containerd` pour les Pods. Containerd applique les paramètres du noyau des nœuds aux espaces de noms des réseaux de Pod. De ce fait, les conteneurs d'un Pod sont liés à la fois aux adresses de bouclage `IPv4` (`127.0.0.1`) et `IPv6` (`::1`). `IPv6` est le

protocole de communication par défaut. Avant de mettre à jour votre cluster vers la version `1.24`, nous vous recommandons de tester vos Pods multi-conteneurs. Modifiez les applications de sorte qu'elles puissent se lier à toutes les adresses IP sur les interfaces de bouclage. La plupart des bibliothèques permettent la liaison `IPv6`, qui est rétrocompatible avec `IPv4`. S'il vous est impossible de modifier le code de votre application, deux options s'offrent à vous :

- Exécutez un conteneur `init` et définissez `disable ipv6` sur `true` (`sysctl -w net.ipv6.conf.all.disable_ipv6=1`).
- Configurez un webhook d'admission en mutation pour injecter un conteneur `init` parallèlement à vos Pods d'applications.

Si vous devez bloquer `IPv6` pour tous les Pods sur tous les nœuds, vous devrez peut-être désactiver `IPv6` sur vos instances.

- L'option `goaway-chance` dans le serveur d'API Kubernetes permet d'éviter que les connexions client `HTTP/2` soient bloquées sur une seule instance de serveur d'API, en fermant une connexion de manière aléatoire. Lorsque la connexion est fermée, le client essaiera de se reconnecter et atterrira probablement sur un autre serveur d'API en raison de l'équilibrage de charge. La version `1.24` de Amazon EKS a activé l'indicateur `goaway-chance`. Si votre charge de travail exécutée sur le cluster Amazon EKS utilise un client qui n'est pas compatible avec `HTTP GOAWAY`, nous vous recommandons de mettre à jour votre client pour qu'il gère `GOAWAY` en se reconnectant à la fin de la connexion.

Pour obtenir le journal complet des modifications de Kubernetes `1.24`, consultez https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.24.md#changelog-since-v1230.

Kubernetes1,23

Kubernetes `1.23` est désormais disponible dans Amazon EKS. Pour plus d'informations sur Kubernetes `1.23`, consultez l'annonce de la version officielle.

> **Important**
>
> - La fonctionnalité de migration de volume de l'interface de stockage de conteneurs (CSI) disponible dans l'arborescence Kubernetes est activée. Cette fonctionnalité permet de remplacer les plugins de stockage existants dans l'arborescence Kubernetes pour Amazon EBS avec un pilote Amazon EBS CSI correspondant. Pour plus d'informations, veuillez consulter la fonctionnalité Kubernetes1.17 : La migration des volumes de l'arborescence Kubernetes vers CSI passe en version bêtasur le blog Kubernetes.

La fonctionnalité traduit les API dans l'arborescence en API CSI équivalentes et délègue les opérations à un pilote CSI de remplacement. Grâce à cette fonctionnalité, si vous utilisez `StorageClass`,`PersistentVolume`, et `PersistentVolumeClaim` qui appartiennent à ces charges de travail, il n'y aura probablement aucun changement conséquent. La fonctionnalité permet à Kubernetes de déléguer toutes les opérations de gestion du stockage depuis le plug-in intégré à l'arborescence au pilote CSI. Si vous utilisez des volumes Amazon EBS dans un cluster existant, vous devrez installer le pilote Amazon EBS CSI avant de mettre à jour votre cluster vers la version `1.23`. Si vous n'installez pas ce pilote avant de mettre à jour un cluster existant, vous pourriez faire face à une interruption de la charge de travail. Si vous prévoyez de déployer des charges de travail qui utilisent des volumes Amazon EBS dans un nouveau cluster `1.23`, installez le pilote Amazon EBS CSI dans votre cluster avant d'y déployer les charges de travail. Pour plus d'instructions sur l'installation du pilote CSI Amazon EBS sur votre cluster, consultez Pilote CSI Amazon EBS. Pour les questions fréquentes sur la fonction de migration, consultez Foire aux questions sur la migration vers Amazon EBS CSI.

- Support étendu pour les Windows AMI optimisées Amazon EKS publiées par AWS n'est pas disponible pour les Kubernetes versions `1.23` mais est disponible pour les Kubernetes versions `1.24` et supérieures.

- Kubernetes ne prend plus en charge `dockershim` dans sa version `1.20` et a supprimé `dockershim` dans la version `1.24`. Pour plus d'informations, voir l'article intitulé Kubernetes is Moving on From Dockershim: Commitments and Next Steps sur le blog Kubernetes. Amazon EKS cessera de prendre en charge `dockershim` à partir de la version `1.24` d'Amazon EKS. À partir de la version `1.24`, les AMI officiels d'Amazon EKS auront `containerd` comme seul environnement d'exécution.

  Même si la version `1.23` d'Amazon EKS continue de prendre en charge `dockershim`, nous vous recommandons de commencer à tester vos applications dès maintenant pour identifier et supprimer les dépendances Docker. En procédant ainsi, vous êtes prêt à mettre à jour votre cluster vers la version `1.24`. Pour en savoir plus sur la suppression de `dockershim`, consultez Amazon EKS a mis fin à la prise en charge de `Dockershim`.
- Kubernetes a fait évolué les `IPv4`/`IPv6` vers la mise en réseau du double empilement Pods, services et nœuds à la disponibilité générale. Toutefois, Amazon EKS et le Amazon VPC CNI plugin for Kubernetes ne prennent actuellement pas en charge la mise en réseau du double

empilement. Vos clusters peuvent attribuer `IPv4` ou `IPv6` adresses de Pods et les services. Toutefois, il ne peut pas attribuer les deux types d'adresses.

- Kubernetes a fait passer la fonctionnalité Pod Security Admission (PSA) en version bêta. Cette fonction est activée par défaut. Pour plus d'informations, consultez Politiques d'admission de sécurité de Pod dans la documentation Kubernetes. PSA remplace le contrôleur d'admission PSP (Pod Security Policy). Le contrôleur d'admission PSP n'est pas pris en charge et sa suppression est prévue dans Kubernetes version`1.25`.

  Le PSP contrôleur d'admission applique Pod normes de sécurité sur Pods dans un espace de noms basé sur des étiquettes d'espace de noms spécifiques qui définissent le niveau d'application. Pour plus d'informations, veuillez consulter normes de sécurité des pods (PSS) et l'admission de sécurité du pod (PSA) dans le Guide des bonnes pratiques Amazon EKS.
- L'image`kube-proxy` déployée avec des clusters est désormais l'image de base minimale gérée par Amazon EKS Distro (EKS-D). L'image contient un minimum de packages et ne possède pas de shell ni de gestionnaire de packages.
- Kubernetes a fait évoluer les conteneurs éphémères vers la version bêta. Les conteneurs éphémères sont des conteneurs temporaires qui s'exécutent dans le même espace de noms qu'un conteneur existant Pod. Vous pouvez les utiliser pour observer l'état de Pods et des conteneurs pour les besoins de dépannage et de débogage. Ceci est particulièrement utile pour le dépannage interactif lorsque `kubectl exec` est insuffisant parce qu'un conteneur a planté ou qu'une image de conteneur n'inclut pas d'utilitaires de débogage. Voici un exemple de conteneur comprenant un utilitaire de débogage :images sans distroless. Pour plus d'informations, veuillez consulter Débogage avec un conteneur de débogage éphémère dans la documentation Kubernetes.
- Kubernetes a fait évoluer l'API stable de `HorizontalPodAutoscaler` `autoscaling/v2` pour une disponibilité générale. L'API `HorizontalPodAutoscaler` `autoscaling/v2beta2` est obsolète. Elle sera indisponible dans `1.26`.
- L'option `goaway-chance` dans le serveur d'API Kubernetes permet d'éviter que les connexions client `HTTP/2` soient bloquées sur une seule instance de serveur d'API, en fermant une connexion de manière aléatoire. Lorsque la connexion est fermée, le client essaiera de se reconnecter et atterrira probablement sur un autre serveur d'API en raison de l'équilibrage de charge. La version `1.23` de Amazon EKS a activé l'indicateur `goaway-chance`. Si votre charge de travail exécutée sur le cluster Amazon EKS utilise un client qui n'est pas compatible avec `HTTP GOAWAY`, nous vous recommandons de mettre à jour votre client pour qu'il gère `GOAWAY` en se reconnectant à la fin de la connexion.

Pour obtenir le journal complet des modifications de Kubernetes `1.23`, consultez la page https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.23.md#changelog-since-v1220.

Notes de mise à jour pour les versions 1.21 et 1.22

> **Important**
>
> Vous ne pouvez pas créer de nouveaux clusters avec ces versions.

Cette rubrique présente les modifications importantes à prendre en compte pour les versions `1.22` et`1.21`. Lors de la mise à niveau, examinez attentivement les modifications apportées entre l'ancienne et la nouvelle version de votre cluster.

Kubernetes version **`1.22`**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version `1.22` :`DefaultStorageClass`,`DefaultTolerationSeconds`,`LimitRanger`,`MutatingAdmissetDefaultIngressClass`.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
`1.22.17`	`eks.28`	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 mai 2024
`1.22.17`	`eks.26`	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	1er avril 2024
`1.22.17`	`eks.14`	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juin 2023

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.22.17	eks.13	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.22.17	eks.12	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 mai 2023
1.22.17	eks.11	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	24 mars 2023
1.22.16	eks.10	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	27 janvier 2023
1.22.15	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 décembre 2022
1.22.15	eks.8	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 novembre 2022
1.22.15	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 novembre 2022
1.22.13	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	21 septembre 2022
1.22.10	eks.5	Nouvelle version de plateforme avec résilience améliorée `etcd`.	15 août 2022

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.22.10	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations. Cette version de plate-forme introduit également un nouveau contrôleur de balisage qui étiquette tous les nœuds de travail avec aws:eks:cluster-name afin de faciliter la répartition des coûts pour ces nœuds de travail. Pour plus d'informations, consultez Identification de vos ressources pour facturation.	21 juillet 2022
1.22.10	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juillet 2022
1.22.9	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	31 mai 2022
1.22.6	eks.1	Version initiale de Kubernetes version 1.22 pour Amazon EKS.	4 avril 2022

Kubernetes version **1.21**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.21 :DefaultStorageClass,DefaultTolerationSeconds,LimitRanger,MutatingAdmissetDefaultIngressClass.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.21.14	eks.33	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 mai 2024
1.21.14	eks.31	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	1er avril 2024
1.21.14	eks.18	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.21.14	eks.17	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 mai 2023
1.21.14	eks.16	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	24 mars 2023
1.21.14	eks.15	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	27 janvier 2023
1.21.14	eks.14	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 décembre 2022
1.21.14	eks.13	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 novembre 2022

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.21.14	eks.12	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 novembre 2022
1.21.13	eks.11	Nouvelle version de plateforme avec résilience améliorée etcd.	10 octobre 2022
1.21.13	eks.10	Nouvelle version de plateforme avec résilience améliorée etcd.	15 août 2022
1.21.13	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations. Cette version de plate-forme introduit également un nouveau contrôleur de balisage qui étiquette tous les nœuds de travail avec aws:eks:cluster-name afin de faciliter la répartition des coûts pour ces nœuds de travail. Pour plus d'informations, consultez Identification de vos ressources pour facturation.	21 juillet 2022
1.21.13	eks.8	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juillet 2022
1.21.12	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	31 mai 2022

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.21.9	eks.6	Le AWS Security Token Service point de terminais on est redevenu le point de terminaison global de la version précédente de la plateforme. Si vous souhaitez utiliser le point de terminaison régional lorsque vous utilisez des rôles IAM pour des comptes de service, vous devez l'activer. Pour obtenir des instructions sur la façon d'activer le point de terminaison régional, consultez Configuration du AWS Security Token Service point de terminaison pour un compte de service.	8 avril 2022
1.21.5	eks.5	Lorsque vous utilisez des Rôles IAM pour les comptes de service, le point de terminais on régional AWS Security Token Service est désormais utilisé par défaut, plutôt que le point de terminaison internati onal. Cette modification est cependant rétablie au point de terminaison international dans eks.6. Un planificateur Fargate mis à jour alimente les nœuds à un taux nettement plus élevé lors de déploiements importants.	10 mars 2022

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.21.5	eks.4	La version 1.10.1-eksbuild.1 du module complémentaire CNI d'Amazon VPC autogéré et Amazon EKS est désormais la version déployée par défaut.	13 décembre 2021
1.21.2	eks.3	Nouvelle version de la plateforme avec prise en charge de la gestion des adresses IPv4 Windows sur le contrôleur de ressources VPC exécuté sur le plan de contrôle Kubernetes. Ajout de la directive de filtre Kubernetes pour la journalisation de Fargate Fluent Bit.	8 novembre 2021
1.21.2	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 septembre 2021
1.21.2	eks.1	Version initiale de Kubernetes version 1.21 pour Amazon EKS.	19 juillet 2021

Versions de la plateforme Amazon EKS

Les versions de la plateforme Amazon EKS représentent les capacités du plan de contrôle du cluster Amazon EKS, telles que les indicateurs de serveur d'API Kubernetes qui sont activés, ainsi que la version de correctif Kubernetes actuelle. Une ou plusieurs versions de plateforme Amazon EKS sont associées à chaque version mineure de Kubernetes. Les versions de plateforme pour les différentes versions mineures de Kubernetes sont indépendantes. Vous pouvez récupérer la version actuelle de la plateforme de votre cluster à l'aide du AWS CLI ou AWS Management Console. Si vous avez un

cluster local activé AWS Outposts, consultez à la Versions de plateforme de clusters locaux Amazon EKS place de cette rubrique.

Lorsqu'une nouvelle version Kubernetes mineure est disponible dans Amazon EKS, telle que la version 1.30, la version initiale de la plateforme Amazon EKS pour cette version Kubernetes mineure commence à`eks.1`. Cependant, Amazon EKS publie régulièrement de nouvelles versions de plateforme pour activer de nouveaux paramètres de plan de contrôle Kubernetes et fournir des correctifs de sécurité.

Lorsque de nouvelles versions de plateforme Amazon EKS deviennent disponibles pour une version mineure :

- Le numéro de version de plateforme Amazon EKS est incrémenté (`eks.`*`n+1`*).
- Amazon EKS met automatiquement à niveau tous les clusters existants vers la dernière version de plateforme Amazon EKS pour les versions mineures Kubernetes correspondantes. Les mises à niveau automatiques de versions de plateforme Amazon EKS existantes sont déployées de façon incrémentielle. Le processus de déploiement peut prendre du temps. Si vous avez besoin immédiatement des fonctions de la dernière version de plateforme Amazon EKS, vous devez créer un nouveau cluster Amazon EKS.

  Si la version de votre cluster est antérieure de plus de deux versions de plateforme par rapport à la version actuelle, il est possible qu'Amazon EKS n'ait pas été en mesure de mettre à jour automatiquement votre cluster. Pour plus de détails sur la cause de ce problème, consultez La version de la plateforme Amazon EKS est inférieure de plus de deux versions à la version actuelle de la plateforme.
- Amazon EKS peut publier une nouvelle AMI de nœud avec une version de correctif correspondante. Toutefois, toutes les versions de correctif sont compatibles entre les AMI de plan de contrôle et de nœud EKS pour une version mineure de Kubernetes donnée.

Les nouvelles versions de plateforme Amazon EKS n'introduisent pas des modifications importantes ou ne provoquent pas d'interruptions de service.

Les clusters sont toujours créés avec la dernière version de plateforme Amazon EKS disponible (`eks.`*`n`*) pour la version Kubernetes spécifiée. Si vous mettez à jour votre cluster vers une nouvelle version mineure de Kubernetes, votre cluster reçoit la version de plateforme Amazon EKS actuelle pour la version mineure Kubernetes vers laquelle vous avez effectué la mise à jour.

Les versions actuelles et récentes de plateforme Amazon EKS sont décrites dans les tableaux ci-dessous.

Kubernetes version `1.30`

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version `1.30` :`NodeRestriction`,`ExtendedResourceToleration`,`NamespaceLifecycle`,`LimitRang`

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
`1.30.1`	`eks.3`	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
`1.30.0`	`eks.2`	Sortie initiale de la version Kubernetes pour EKS. `1.30` Pour plus d'informations, consultez Kubernetes1,30.	23 mai 2024

Kubernetes version `1.29`

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version `1.29` :`NodeRestriction`,`ExtendedResourceToleration`,`NamespaceLifecycle`,`LimitRang`

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
`1.29.5`	`eks.8`	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
`1.29.4`	`eks.7`	Nouvelle version de la plateforme avec mise à l'échelle automatique de CoreDNS, correctifs de sécurité et améliorations. Pour plus d'informations sur la mise à l'échelle automatiq	16 mai 2024

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
		ue de CoreDNS, consultez. Mise à l'échelle automatique CoreDNS	
1.29.3	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.29.1	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024
1.29.1	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.29.1	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.29.0	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.29 Pour plus d'informations, consultez Kubernetes1,29.	23 janvier 2024

Kubernetes version **1.28**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.28 :NodeRestriction,ExtendedResourceToleration,NamespaceLifecycle,LimitRang

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.28.10	eks.14	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.28.9	eks.13	Nouvelle version de la plateforme avec mise à l'échelle automatique de CoreDNS, correctifs de sécurité et améliorations. Pour plus d'informations sur la mise à l'échelle automatique de CoreDNS, consultez. Mise à l'échelle automatique CoreDNS	16 mai 2024
1.28.8	eks.12	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.28.7	eks.11	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024
1.28.7	eks.10	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.28.6	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.28.5	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 janvier 2024
1.28.4	eks.6	Nouvelle version de plateforme avec des entrées d'accès, des corrections de sécurité et des améliorations.	14 décembre 2023
1.28.4	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 décembre 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.28.3	eks.4	Nouvelle version de plateforme avec des Identités du pod EKS, des corrections de sécurité et des améliorations.	10 novembre 2023
1.28.3	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	3 novembre 2023
1.28.2	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 octobre 2023
1.28.1	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.28 Pour plus d'informations, consultez Kubernetes1,28.	26 septembre 2023

Kubernetes version **1.27**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.27 :NodeRestriction,ExtendedResourceToleration,NamespaceLifecycle,LimitRang

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.27.14	eks.18	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
1.27.13	eks.17	Nouvelle version de la plateforme avec mise à l'échelle automatique de CoreDNS, correctifs de sécurité et améliorations. Pour plus d'informa	16 mai 2024

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
		tions sur la mise à l'échelle automatique de CoreDNS, consultez. Mise à l'échelle automatique CoreDNS	
1.27.12	eks.16	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.27.11	eks.15	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024
1.27.11	eks.14	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.27.10	eks.13	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.27.9	eks.11	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 janvier 2024
1.27.8	eks.10	Nouvelle version de plateforme avec des entrées d'accès, des corrections de sécurité et des améliorations.	14 décembre 2023
1.27.8	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 décembre 2023
1.27.7	eks.8	Nouvelle version de plateforme avec des Identités du pod EKS, des corrections de sécurité et des améliorations.	10 novembre 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.27.7	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	3 novembre 2023
1.27.6	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 octobre 2023
1.27.4	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 août 2023
1.27.4	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juillet 2023
1.27.3	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juin 2023
1.27.2	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.27.1	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.27 Pour plus d'informations, consultez Kubernetes1,27.	24 mai 2023

Kubernetes version **1.26**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.26 :NodeRestriction,ExtendedResourceToleration,NamespaceLifecycle,LimitRang

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.26.15	eks.19	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
1.26.15	eks.18	Nouvelle version de la plateforme avec mise à l'échelle automatique de CoreDNS, correctifs de sécurité et améliorations. Pour plus d'informations sur la mise à l'échelle automatique de CoreDNS, consultez. Mise à l'échelle automatique CoreDNS	16 mai 2024
1.26.15	eks.17	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.26.14	eks.16	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024
1.26.14	eks.15	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.26.13	eks.14	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.26.12	eks.12	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 janvier 2024
1.26.11	eks.11	Nouvelle version de plateforme avec des entrées d'accès, des corrections de sécurité et des améliorations.	14 décembre 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.26.11	eks.10	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 décembre 2023
1.26.10	eks.9	Nouvelle version de plateforme avec des Identités du pod EKS, des corrections de sécurité et des améliorations.	10 novembre 2023
1.26.10	eks.8	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	3 novembre 2023
1.26.9	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 octobre 2023
1.26.7	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 août 2023
1.26.7	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juillet 2023
1.26.6	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juin 2023
1.26.5	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.26.4	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 mai 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.26.2	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.26 Pour plus d'informations, consultez Kubernetes1,26.	11 avril 2023

Kubernetes version **1.25**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.25 :NodeRestriction,ExtendedResourceToleration,NamespaceLifecycle,LimitRang

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.25.16	eks.20	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
1.25.16	eks.19	Nouvelle version de la plateforme avec mise à l'échelle automatique de CoreDNS, correctifs de sécurité et améliorations. Pour plus d'informations sur la mise à l'échelle automatique de CoreDNS, consultez. Mise à l'échelle automatique CoreDNS	16 mai 2024
1.25.16	eks.18	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.25.16	eks.17	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.25.16	eks.16	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.25.16	eks.15	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.25.16	eks.13	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 janvier 2024
1.25.16	eks.12	Nouvelle version de plateforme avec des entrées d'accès, des corrections de sécurité et des améliorations.	14 décembre 2023
1.25.16	eks.11	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 décembre 2023
1.25.15	eks.10	Nouvelle version de plateforme avec des Identités du pod EKS, des corrections de sécurité et des améliorations.	10 novembre 2023
1.25.15	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	3 novembre 2023
1.25.14	eks.8	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 octobre 2023
1.25.12	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 août 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.25.12	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juillet 2023
1.25.11	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juin 2023
1.25.10	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.25.9	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 mai 2023
1.25.8	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	24 mars 2023
1.25.6	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.25 Pour plus d'informations, consultez Kubernetes1,25.	21 février 2023

Kubernetes version **1.24**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.24 :CertificateApproval,CertificateSigning,CertificateSubjectRestriction,DefaultStorageClass,DefaultTolerationSeconds,LimitRanger,MutatingAdmissionWebhook,NamespaceLifecycle,NodeRestriction,PersistentVolumeClaimResize,Priority,ResourceQuota,RuntimeClass,ServiceAccount,StorageObjectInUseProtection,TaintNodesByCondition,ValidatingAdmissionWebhook.

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.24.17	eks.23	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
1.24.17	eks.22	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 mai 2024
1.24.17	eks.21	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.24.17	eks.20	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024
1.24.17	eks.19	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.24.17	eks.18	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.24.17	eks.16	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 janvier 2024
1.24.17	eks.15	Nouvelle version de plateforme avec des entrées d'accès, des corrections de sécurité et des améliorations.	14 décembre 2023
1.24.17	eks.14	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 décembre 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.24.17	eks.13	Nouvelle version de plateforme avec des Identités du pod EKS, des corrections de sécurité et des améliorations.	10 novembre 2023
1.24.17	eks.12	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	3 novembre 2023
1.24.17	eks.11	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 octobre 2023
1.24.16	eks.10	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 août 2023
1.24.16	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juillet 2023
1.24.15	eks.8	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juin 2023
1.24.14	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.24.13	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 mai 2023
1.24.12	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	24 mars 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.24.8	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	27 janvier 2023
1.24.7	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 décembre 2022
1.24.7	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 novembre 2022
1.24.7	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.24 Pour plus d'informations, consultez Kubernetes1,24.	15 novembre 2022

Kubernetes version **1.23**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.23 :CertificateApproval,CertificateSigning,CertificateSubjectRestriction,DefaultStorageClass,DefaultTolerationSeconds,LimitRanger,MutatingAdmissionWebhook,NamespaceLifecycle,NodeRestriction,PersistentVolumeClaimResize,Priority,ResourceQuota,RuntimeClass,ServiceAccount,StorageObjectInUseProtection,ValidatingAdmissionWebhook.

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.23.17	eks.25	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 juin 2024
1.23.17	eks.24	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 mai 2024

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.23.17	eks.23	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 avril 2024
1.23.17	eks.22	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	29 mars 2024
1.23.17	eks.21	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	20 mars 2024
1.23.17	eks.20	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 mars 2024
1.23.17	eks.18	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	17 janvier 2024
1.23.17	eks.17	Nouvelle version de plateforme avec des entrées d'accès, des corrections de sécurité et des améliorations.	14 décembre 2023
1.23.17	eks.16	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	12 décembre 2023
1.23.17	eks.15	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	10 novembre 2023
1.23.17	eks.14	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	3 novembre 2023

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.23.17	eks.13	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	16 octobre 2023
1.23.17	eks.12	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 août 2023
1.23.17	eks.11	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juillet 2023
1.23.17	eks.10	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	30 juin 2023
1.23.17	eks.9	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	9 juin 2023
1.23.17	eks.8	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 mai 2023
1.23.17	eks.7	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	24 mars 2023
1.23.14	eks.6	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	27 janvier 2023
1.23.13	eks.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	5 décembre 2022

Version de Kubernetes	Version de la plateforme EKS	Notes de mise à jour	Date de publication
1.23.13	eks.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	18 novembre 2022
1.23.12	eks.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	7 novembre 2022
1.23.10	eks.2	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	21 septembre 2022
1.23.7	eks.1	Sortie initiale de la version Kubernetes pour EKS. 1.23 Pour plus d'informations, consultez Kubernetes1,23.	11 août 2022

Obtenir la version actuelle de la plateforme

Pour obtenir la version actuelle de la plateforme pour votre cluster (console)

1. Ouvrez la console Amazon EKS.
2. Dans le panneau de navigation, choisissez Clusters.
3. Dans la liste des clusters, choisissez le nom du cluster pour vérifier la version de la plateforme.
4. Choisissez l'onglet Overview (Présentation).
5. La version de la plateforme est disponible dans la section Détails.

Pour obtenir la version actuelle de la plate-forme pour votre cluster (AWS CLI)

1. Déterminez le nom du cluster dont vous souhaitez vérifier la version de la plateforme.
2. Exécutez la commande suivante :

```
aws eks describe-cluster --name my-cluster --query cluster.platformVersion
```

L'exemple qui suit illustre un résultat.

```
"eks.10"
```

Autoscaling

La scalabilité automatique est une fonction qui augmente ou diminue automatiquement vos ressources en fonction de l'évolution de la demande. Il s'agit d'une fonction majeure de Kubernetes qui, autrement, nécessiterait des ressources humaines importantes pour être exécutée manuellement.

Amazon EKS prend en charge deux produits d'autoscaling :

Karpenter

Karpenter est un outil Cluster Autoscaler de Kubernetes flexible hautes performances qui permet d'améliorer la disponibilité des applications et l'efficacité du cluster. Karpenter lance des ressources de calcul de taille appropriée (par exemple, des instances Amazon EC2) en réponse à l'évolution de la charge de l'application en moins d'une minute. Grâce à l'intégration de Kubernetes à AWS, Karpenter peut fournir des ressources de calcul juste à temps qui répondent précisément aux exigences de votre charge de travail. Karpenteralloue automatiquement de nouvelles ressources de calcul en fonction des exigences spécifiques des charges de travail du cluster. Celles-ci comprennent les exigences en matière de calcul, de stockage, d'accélération et de planification. Amazon EKS prend en charge les clusters utilisant Karpenter, bien que Karpenter fonctionne avec tout cluster Kubernetes conforme. Pour de plus amples informations, veuillez consulter la documentation Karpenter.

Cluster Autoscaler

Le Cluster Autoscaler de Kubernetes ajuste automatiquement le nombre de nœuds dans votre cluster lorsque les pods échouent ou sont reprogrammés sur d'autres nœuds. Le Cluster Autoscaler utilise des groupes Auto Scaling. Pour plus d'informations, consultez Cluster Autoscaler sur AWS.

Gérez l'accès

Découvrez comment gérer l'accès à votre cluster Amazon EKS. L'utilisation d'Amazon EKS nécessite de savoir comment Kubernetes et AWS Identity and Access Management (AWS IAM) gèrent le contrôle d'accès.

Cette section inclut :

the section called “Autoriser l'accès aux API Kubernetes”— Découvrez comment permettre aux applications ou aux utilisateurs de s'authentifier auprès de l'KubernetesAPI. Vous pouvez utiliser des entrées d'accès, l'aws-auth ou un ConfigMap fournisseur OIDC externe.

the section called “Accéder à mon cluster avec kubectl”— Découvrez comment configurer kubectl pour communiquer avec votre cluster Amazon EKS. Utilisez la AWS CLI pour créer un fichier kubeconfig.

the section called “Accordez aux charges de travail l'accès à AWS”— Découvrez comment associer un compte de Kubernetes service à AWS IAM Roles. Vous pouvez utiliser Pod Identity ou IAM Roles for Service Accounts (IRSA).

Tâches courantes :

- Accordez aux développeurs l'accès à l'KubernetesAPI. Consultez Kubernetes les ressources dans le AWS Management Console.
  - Solution : utilisez les entrées d'accès pour associer les autorisations Kubernetes RBAC aux utilisateurs ou aux rôles AWS IAM.
- Configurez kubectl pour communiquer avec un cluster Amazon EKS à l'aide AWS des informations d'identification.
  - Solution : utilisez la AWS CLI pour créer un fichier kubeconfig.
- Utilisez un fournisseur d'identité externe, tel que Ping Identity, pour authentifier les utilisateurs auprès de l'KubernetesAPI.
  - Solution : associez un fournisseur OIDC externe.
- Donnez aux charges de travail de votre Kubernetes cluster la possibilité d'appeler des AWS API.
  - Solution : utilisez Pod Identity pour associer un rôle AWS IAM à un compte Kubernetes de service.

Contexte :

- Découvrez comment fonctionnent les comptes de service Kubernetes.
- Passez en revue le modèle de contrôle d'accès basé sur les rôles (RBAC) de Kubernetes
- Pour plus d'informations sur la gestion de l'accès aux AWS ressources, consultez le guide de l'utilisateur AWS IAM. Vous pouvez également suivre une formation d'introduction gratuite sur l'utilisation d' AWS IAM.

Autoriser l'accès aux Kubernetes API

Votre cluster possède un point de terminaison d'KubernetesAPI. Kubectl utilise cette API. Vous pouvez vous authentifier auprès de cette API à l'aide de deux types d'identités :

- Un principal AWS Identity and Access Management (IAM) (rôle ou utilisateur) : ce type nécessite une authentification auprès d'IAM. Les utilisateurs peuvent se connecter en AWS tant qu'utilisateur IAM ou avec une identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. Les utilisateurs ne peuvent se connecter qu'avec une identité fédérée si votre administrateur a précédemment configuré une fédération d'identité avec des rôles IAM. Lorsque les utilisateurs accèdent AWS en utilisant la fédération, ils assument indirectement un rôle. Lorsque les utilisateurs utilisent ce type d'identité :
  - Vous pouvez leur attribuer des autorisations Kubernetes afin qu'ils puissent travailler avec des objets Kubernetes de votre cluster. Pour plus d'informations sur la façon d'attribuer des autorisations à vos principaux IAM afin qu'ils puissent accéder aux objets Kubernetes de votre cluster, consultez Gérer les entrées d'accès
  - Vous pouvez leur attribuer des autorisations IAM afin qu'ils puissent travailler avec votre cluster Amazon EKS et ses ressources à l'aide de l'API Amazon EKS AWS CLI, AWS CloudFormation, AWS Management Console, ou`eksctl`. Pour plus d'informations, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service dans la Référence des autorisations de service.
  - Les nœuds rejoignent votre cluster en endossant un rôle IAM. La capacité à accéder à votre cluster à l'aide des principaux IAM est fournie par l'authentificateur IAM AWS pour Kubernetes, qui s'exécute sur le plan de contrôle Amazon EKS.
- Un utilisateur de votre propre fournisseur OpenID Connect (OIDC) : ce type nécessite une authentification auprès de votre fournisseur OIDC. Pour de plus amples informations sur la configuration de votre propre fournisseur OIDC avec votre cluster Amazon EKS, veuillez consulter Authentifier les utilisateurs de votre cluster auprès d'un fournisseur d'OpenID Connectidentité. Lorsque les utilisateurs utilisent ce type d'identité :

- Vous pouvez leur attribuer des autorisations Kubernetes afin qu'ils puissent travailler avec des objets Kubernetes de votre cluster.
- Impossible de leur attribuer des autorisations IAM afin qu'ils puissent travailler avec votre cluster Amazon EKS et ses ressources à l'aide de l'API Amazon EKS AWS CLI,, AWS CloudFormation AWS Management Console, ou`eksctl`.

Vous pouvez utiliser les deux types d'identités avec votre cluster. La méthode d'authentification IAM ne peut pas être désactivée. La méthode d'authentification OIDC est facultative.

Associer les identités IAM aux autorisations Kubernetes

L'authentificateur AWS IAM pour Kubernetes est installé sur le plan de contrôle de votre cluster. Il permet aux principaux (rôles et utilisateurs) AWS Identity and Access Management (IAM) que vous autorisez à accéder aux ressources Kubernetes de votre cluster. Vous pouvez autoriser les principaux IAM à accéder aux objets Kubernetes de votre cluster en utilisant l'une des méthodes suivantes :

- Création d'entrées d'accès : si votre cluster correspond à la version de plateforme répertoriée dans la section Conditions requises pour la version Kubernetes de votre cluster ou est ultérieure à celle-ci, nous vous recommandons d'utiliser cette option.

  Utilisez les entrées d'accès pour gérer les autorisations Kubernetes des principaux IAM depuis l'extérieur du cluster. Vous pouvez ajouter et gérer l'accès au cluster à l'aide de l'API AWS EKS AWS Command Line Interface AWS CloudFormation, des SDK et AWS Management Console. Cela signifie que vous pouvez gérer les utilisateurs avec les mêmes outils que ceux avec lesquels vous avez créé le cluster.

  Pour commencer, suivez Configuration des entrées d'accès, puis Migration des entrées `aws-auth` `ConfigMap` existantes pour accéder aux entrées.
- Ajouter des entrées à la **`ConfigMapaws-auth`** : si la version de la plateforme de votre cluster est antérieure à la version répertoriée dans la section Conditions préalables, vous devez utiliser cette option. Si la version de plateforme de votre cluster est identique ou ultérieure à la version de plateforme répertoriée dans la section Conditions requises pour la version Kubernetes de votre cluster et que vous avez ajouté des entrées à la `ConfigMap`, nous vous recommandons de migrer ces entrées vers des entrées d'accès. Vous ne pouvez toutefois pas migrer les entrées qu'Amazon EKS a ajoutées à la `ConfigMap`, telles que les entrées pour les rôles IAM utilisés

avec des groupes de nœuds gérés ou des profils Fargate. Pour plus d'informations, consultez the section called "Autoriser l'accès aux API Kubernetes".

- Si vous devez utiliser l'option `ConfigMap aws-auth`, vous pouvez ajouter des entrées à la `ConfigMap` à l'aide de la commande **`eksctl create iamidentitymapping`**. Pour plus d'informations, consultez la section Gérer les utilisateurs et les rôles IAM dans la documentation `eksctl`.

Définir le mode d'authentification du cluster

Chaque cluster possède un mode d'authentification. Le mode d'authentification détermine les méthodes que vous pouvez utiliser pour autoriser les principaux IAM à accéder aux objets Kubernetes de votre cluster. Il existe trois modes d'authentification.

> **Important**
>
> Une fois que la méthode de saisie d'accès est activée, elle ne peut pas être désactivée. Si la `ConfigMap` méthode n'est pas activée lors de la création du cluster, elle ne pourra pas être activée ultérieurement. La `ConfigMap` méthode est activée pour tous les clusters créés avant l'introduction des entrées d'accès.

La `ConfigMap aws-auth` au sein du cluster

Il s'agit du mode d'authentification d'origine pour les clusters Amazon EKS. Le principal IAM qui a créé le cluster est l'utilisateur initial à avoir accès au cluster au moyen de `kubectl`. L'utilisateur initial doit ajouter d'autres utilisateurs à la liste dans la `ConfigMap aws-auth` et attribuer des autorisations qui affectent les autres utilisateurs au sein du cluster. Ces autres utilisateurs ne peuvent pas gérer ou supprimer l'utilisateur initial, car il n'y a aucune entrée à gérer dans la `ConfigMap`.

À la fois la `ConfigMap` et les entrées d'accès

Avec ce mode d'authentification, vous pouvez utiliser les deux méthodes pour ajouter des principaux IAM au cluster. Notez que chaque méthode stocke des entrées distinctes ; par exemple, si vous ajoutez une entrée d'accès depuis le AWS CLI, elle n'`aws-auth`est `ConfigMap`pas mise à jour.

Accès aux entrées uniquement

Avec ce mode d'authentification, vous pouvez utiliser l'API AWS EKS AWS Command Line Interface AWS CloudFormation, les SDK et gérer l'accès AWS Management Console au cluster pour les principaux IAM.

Chaque entrée d'accès possède un type et vous pouvez utiliser la combinaison d'une portée d'accès pour limiter le principal à un espace de noms spécifique et d'une stratégie d'accès pour définir des politiques d'autorisations réutilisables préconfigurées. Vous pouvez également utiliser le type STANDARD et les groupes Kubernetes RBAC pour attribuer des autorisations personnalisées.

Mode d'authentification	Méthodes
`ConfigMap` uniquement (`CONFIG_MAP`)	`aws-auth ConfigMap`
API EKS et `ConfigMap` (`API_AND_CONFIG_MAP`)	accéder aux entrées de l'API AWS EKS AWS Command Line Interface, des SDK AWS CloudFormation, et AWS Management Console `aws-auth ConfigMap`
API EKS uniquement (`API`)	accéder aux entrées de l'API AWS EKS AWS Command Line Interface, des SDK AWS CloudFormation, et AWS Management Console

Gérer les entrées d'accès

Prérequis

- Découverte des options d'accès au cluster pour votre cluster Amazon EKS. Pour plus d'informations, consultez Autoriser l'accès aux Kubernetes API .
- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS. Pour utiliser les entrées d'accès et modifier le mode d'authentification d'un cluster, celui-ci doit avoir une version de plateforme identique ou ultérieure à la version répertoriée dans le tableau suivant, ou une version Kubernetes postérieure aux versions répertoriées dans le tableau.

Version de Kubernetes	Version de plateforme
1.30	eks.2
1.29	eks.1
1.28	eks.6
1.27	eks.10
1.26	eks.11
1.25	eks.12
1.24	eks.15
1.23	eks.17

Vous pouvez vérifier la version actuelle de votre Kubernetes et la version de plateforme en remplaçant *my-cluster* dans la commande suivante par le nom de votre cluster et en exécutant la commande modifiée : **aws eks describe-cluster --name *my-cluster* --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'**.

> **Important**
>
> Une fois qu'Amazon EKS a mis à jour votre cluster vers la version de plateforme répertoriée dans le tableau, Amazon EKS crée une entrée d'accès avec des autorisations d'administrateur au cluster pour le principal IAM qui a initialement créé le cluster. Si vous ne souhaitez pas que ce principal IAM dispose d'autorisations d'administrateur sur le cluster, supprimez l'entrée d'accès créée par Amazon EKS.
> Pour les clusters dont les versions de plateforme sont antérieures à celles répertoriées dans le tableau précédent, le créateur du cluster est toujours un administrateur de cluster. Il n'est pas possible de supprimer les autorisations d'administrateur de cluster du rôle IAM ou de l'utilisateur IAM qui a créé le cluster.

- Un principal IAM disposant des autorisations suivantes pour votre cluster :`CreateAccessEntry`, `ListAccessEntries`, `DescribeAccessEntry`, `DeleteAccessEntry` et `UpdateAccessEntry`. Pour plus d'informations sur les autorisations Amazon EKS, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service dans la Référence des autorisations de service.
- Un principal IAM existant pour lequel créer une entrée d'accès, ou une entrée d'accès existante à mettre à jour ou à supprimer.

Configuration des entrées d'accès

Pour commencer à utiliser les entrées d'accès, vous devez remplacer le mode d'authentification du cluster par les modes `API_AND_CONFIG_MAP` ou `API`. Cela ajoute l'API pour les entrées d'accès.

AWS Management Console

Pour créer une entrée d'accès

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster pour lequel vous souhaitez créer une entrée d'accès.
3. Choisissez l'onglet Access.
4. Le mode d'authentification indique le mode d'authentification actuel du cluster. Si le mode indique EKS API, vous pouvez déjà ajouter des entrées d'accès et vous pouvez ignorer les étapes restantes.
5. Choisissez Gérer l'accès.
6. Pour le mode d'authentification du cluster, sélectionnez un mode avec l'EKS API. Notez que vous ne pouvez pas revenir à un mode d'authentification qui supprime l'EKS API et les entrées d'accès.
7. Sélectionnez Enregistrer les modifications. Amazon EKS commence à mettre à jour le cluster, le statut du cluster passe à Updating et le changement est enregistré dans l'onglet Historique de mise à jour.
8. Attendez que le statut du cluster revienne à Active. Lorsque le cluster est Active, vous pouvez suivre les étapes décrites dans Création d'entrées d'accès pour ajouter l'accès au cluster pour les principaux IAM.

AWS CLI

Prérequis

La dernière version de la AWS CLI v1 installée et configurée sur votre appareil ou AWS CloudShell. AWS CLI La v2 ne prend pas en charge les nouvelles fonctionnalités depuis quelques jours. Vous pouvez vérifier votre version actuelle avec `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, reportez-vous à la section Installation, mise à jour et désinstallation de la AWS CLI configuration rapide `aws configure` dans le guide de l' AWS Command Line Interface utilisateur. La AWS CLI version installée dans le AWS CloudShell peut également être en retard de plusieurs versions par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

1.

2. Exécutez la commande suivante. Remplacez *my-cluster* par le nom de votre cluster. Si vous souhaitez désactiver définitivement la méthode `ConfigMap`, remplacez `API_AND_CONFIG_MAP` par `API`.

   Amazon EKS commence à mettre à jour le cluster, le statut du cluster passe à UPDATING et le changement est enregistré dans aws eks list-updates.

   ```
   aws eks update-cluster-config --name my-cluster --access-config
    authenticationMode=API_AND_CONFIG_MAP
   ```

3. Attendez que le statut du cluster revienne à Active. Lorsque le cluster est Active, vous pouvez suivre les étapes décrites dans Création d'entrées d'accès pour ajouter l'accès au cluster pour les principaux IAM.

Création d'entrées d'accès

Considérations

Avant de créer des entrées d'accès, tenez compte des éléments suivants :

- Une entrée d'accès inclut l'Amazon Resource Name (ARN) d'un et d'un seul principal IAM existant. Un principal IAM ne peut pas être inclus dans plus d'une entrée d'accès. Considérations supplémentaires concernant l'ARN que vous spécifiez :
  - Les bonnes pratiques IAM recommandent d'accéder à votre cluster en utilisant des rôles IAM dotés d'informations d'identification à court terme, plutôt que des utilisateurs IAM dotés d'informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le guide de l'utilisateur IAM.
  - Si l'ARN est destiné à un rôle IAM, il peut inclure un chemin. Les ARN dans les entrées `ConfigMap aws-auth` ne peuvent pas inclure de chemin. Par exemple, votre ARN peut être `arn:aws:iam::111122223333:role/development/apps/my-role` ou`arn:aws:iam::111122223333:role/my-role`.
  - Si le type de l'entrée d'accès est autre que `STANDARD` (voir la prochaine considération concernant les types), l'ARN doit être identique à celui dans Compte AWS lequel se trouve votre cluster. Si c'est le cas`STANDARD`, l'ARN peut être identique ou Compte AWS différent du compte dans lequel se trouve votre cluster.
  - Vous ne pouvez pas modifier le principal IAM une fois l'entrée d'accès créée.
  - Si vous supprimez le principal IAM avec cet ARN, l'entrée d'accès n'est pas automatiquement supprimée. Nous vous recommandons de supprimer l'entrée d'accès avec un ARN pour un principal IAM que vous supprimez. Si vous ne supprimez pas l'entrée d'accès et que vous recréez le principal IAM, même s'il possède le même ARN, l'entrée d'accès ne fonctionnera pas. En effet, même si l'ARN est le même pour le principal IAM recréé, le `roleID` or `userID` (vous pouvez le constater avec la `aws sts get-caller-identity` AWS CLI commande) est différent pour le principal IAM recréé de ce qu'il était pour le principal IAM d'origine. Même si vous ne voyez pas l'`roleID` ou l'`userID` du principal IAM d'une entrée d'accès, Amazon EKS la stocke avec l'entrée d'accès.
- Chaque entrée d'accès possède un type. Vous pouvez spécifier `EC2 Linux` (pour un rôle IAM utilisé avec des nœuds autogérés Linux ou Bottlerocket), `EC2 Windows` (pour un rôle IAM utilisé avec des nœuds autogérés Windows), `FARGATE_LINUX` (pour un rôle IAM utilisé avec) ou en tant que type. AWS Fargate (Fargate)`STANDARD` Si vous ne spécifiez pas de type, Amazon EKS définit automatiquement le type sur `STANDARD`. Il n'est pas nécessaire de créer une entrée d'accès pour un rôle IAM utilisé pour un groupe de nœuds gérés ou un profil Fargate, car Amazon EKS ajoute des entrées pour ces rôles à la `ConfigMap aws-auth`, quelle que soit la version de plateforme utilisée par votre cluster.

Vous ne pouvez pas modifier le type une fois l'entrée d'accès créée.

- Si le type d'entrée d'accès est `STANDARD`, vous pouvez spécifier un nom d'utilisateur pour l'entrée d'accès. Si vous ne spécifiez aucune valeur pour le nom d'utilisateur, Amazon EKS définit l'une des valeurs suivantes pour vous, en fonction du type d'entrée d'accès et du fait que le principal IAM que vous avez spécifié est un rôle IAM ou un utilisateur IAM. À moins que vous n'ayez une raison précise de spécifier votre propre nom d'utilisateur, nous vous recommandons de ne pas en indiquer et de laisser Amazon EKS le générer automatiquement pour vous. Si vous spécifiez votre propre nom d'utilisateur :
  - Il ne peut pas commencer par `system:`, `eks:`, `aws:`, `amazon:` ou `iam:`.
  - Si le nom d'utilisateur correspond à un rôle IAM, nous vous recommandons d'ajouter `{{SessionName}}` à la fin de votre nom d'utilisateur. Si vous ajoutez un nom d'utilisateur `{{SessionName}}` à votre nom d'utilisateur, celui-ci doit inclure deux points avant {{SessionName}}. Lorsque ce rôle est assumé, le nom de la session spécifiée lors de l'attribution du rôle est automatiquement transmis au cluster et apparaît dans CloudTrail les journaux. Par exemple, vous ne pouvez pas avoir le nom d'utilisateur `john{{SessionName}}`. Le nom d'utilisateur doit être `:john{{SessionName}}` ou `jo:hn{{SessionName}}`. Il suffit que les deux-points soient devant `{{SessionName}}`. Le nom d'utilisateur généré par Amazon EKS dans le tableau suivant inclut un ARN. Comme un ARN inclut les deux-points, il répond à cette exigence. Les deux-points ne sont pas obligatoires si vous n'incluez pas `{{SessionName}}` dans votre nom d'utilisateur.

Type du principal IAM	Type	Valeur du nom d'utilisateur définie automatiquement par Amazon EKS
Utilisateur	`STANDARD`	ARN de l'utilisateur IAM. Exemple : `arn:aws:iam::` *`111122223333`* `:user/`*`my-user`*
Rôle	`STANDARD`	L'ARN STS du rôle lorsqu'il est assumé. Amazon EKS ajoute `{{SessionName}}` au rôle.

Type du principal IAM	Type	Valeur du nom d'utilisateur définie automatiquement par Amazon EKS
		Exemple : `arn:aws:sts::` *`111122223333`* `:assumed-role/` *`my-role`*`/{{SessionName}}` Si l'ARN du rôle que vous avez spécifié contenait un chemin, Amazon EKS le supprime dans le nom d'utilisateur généré.
Rôle	`EC2 Linux` ou `EC2 Windows`	`system:node:{{EC2PrivateDNSName}}`
Rôle	`FARGATE_LINUX`	`system:node:{{SessionName}}`

Vous ne pouvez pas modifier le nom d'utilisateur une fois l'entrée d'accès créée.

- Si le type d'entrée d'accès est `STANDARD` et que vous souhaitez utiliser l'autorisation Kubernetes RBAC, vous pouvez ajouter un ou plusieurs noms de groupe à l'entrée d'accès. Après avoir créé une entrée d'accès, vous pouvez ajouter et supprimer des noms de groupes. Pour que le principal IAM ait accès aux objets Kubernetes de votre cluster, vous devez créer et gérer des objets d'autorisation basée sur les rôles (RBAC) Kubernetes. Créez des objets Kubernetes `RoleBinding` ou `ClusterRoleBinding` sur votre cluster qui spécifient le nom du groupe sous forme de `subject` pour `kind: Group`. Kubernetes autorise le principal IAM à accéder à tous les objets de cluster que vous avez spécifiés dans un objet Kubernetes `Role` ou `ClusterRole` que vous avez également spécifié dans `roleRef` de votre liaison. Si vous spécifiez des noms de groupe, nous vous recommandons de bien connaître les objets d'autorisation basée sur les rôles (RBAC) Kubernetes. Pour plus d'informations, consultez Utilisation de l'autorisation RBAC dans la documentation Kubernetes.

> **Important**
>
> Amazon EKS ne confirme pas que les objets RBAC Kubernetes qui existent sur votre cluster incluent les noms de groupe que vous spécifiez.

Au lieu, ou en plus, de Kubernetes autorisant l'accès principal IAM aux objets Kubernetes de votre cluster, vous pouvez associer les politiques d'accès Amazon EKS à une entrée d'accès. Amazon EKS autorise les principaux IAM à accéder aux objets Kubernetes de votre cluster avec les autorisations définies dans la stratégie d'accès. Vous pouvez étendre les autorisations d'une stratégie d'accès aux espaces de noms Kubernetes que vous spécifiez. L'utilisation de stratégies d'accès ne vous oblige pas à gérer les objets RBAC Kubernetes. Pour plus d'informations, consultez Associer les stratégies d'accès aux entrées d'accès et les dissocier des entrées d'accès.

- Si vous créez une entrée d'accès avec le type `EC2 Linux` ou `EC2 Windows`, le principal IAM qui crée l'entrée d'accès doit avoir l'autorisation `iam:PassRole`. Pour plus d'informations, consultez Octroi d'autorisations à un utilisateur pour transférer un rôle à un Service AWS dans le Guide de l'utilisateur IAM.
- À l'instar du comportement IAM standard, la création et les mises à jour des entrées d'accès sont finalement cohérentes et prennent effet au bout de plusieurs secondes après que l'appel API initial a été renvoyé avec succès. Vous devez concevoir vos applications de sorte qu'elles tiennent compte de ces retards potentiels. Nous vous recommandons de ne pas inclure les créations ou mises à jour des entrées d'accès dans les chemins de code critique et haute disponibilité de votre application. Au lieu de cela, procédez aux modifications dans une routine d'initialisation ou d'installation distincte que vous exécutez moins souvent. Veillez également à vérifier que les modifications ont été propagées avant que les processus de production en dépendent.
- Les entrées d'accès ne prennent pas en charge les rôles liés aux services. Vous ne pouvez pas créer d'entrées d'accès dont l'ARN principal est un rôle lié à un service. Vous pouvez identifier les rôles liés à un service par leur ARN, qui est au format `arn:aws:iam::*:role/aws-service-role/*`.

Vous pouvez créer une entrée d'accès à l'aide du AWS Management Console ou du AWS CLI.

AWS Management Console

Pour créer une entrée d'accès

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster pour lequel vous souhaitez créer une entrée d'accès.
3. Choisissez l'onglet Access.
4. Choisissez Créer une entrée d'accès.
5. Pour le principal IAM, sélectionnez un utilisateur ou un rôle IAM existant. Les bonnes pratiques IAM recommandent d'accéder à votre cluster en utilisant des rôles IAM dotés d'informations d'identification à court terme, plutôt que des utilisateurs IAM dotés d'informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le guide de l'utilisateur IAM.
6. Pour Type, si l'entrée d'accès concerne le rôle de nœud utilisé pour les nœuds Amazon EC2 autogérés, sélectionnez EC2 Linux ou EC2 Windows. Dans le cas contraire, acceptez le type par défaut (Standard).
7. Si le type que vous avez choisi est Standard et que vous souhaitez spécifier un nom d'utilisateur, saisissez-le.
8. Si le type que vous avez choisi est Standard et que vous souhaitez utiliser l'autorisation Kubernetes RBAC pour le principal IAM, spécifiez un ou plusieurs noms pour Groupes. Si vous ne spécifiez aucun nom de groupe et que vous souhaitez utiliser l'autorisation Amazon EKS, vous pouvez associer une stratégie d'accès ultérieurement ou une fois l'entrée d'accès créée.
9. (Facultatif) Pour Balises, attribuez des étiquettes à l'entrée d'accès. Par exemple, pour faciliter la recherche de toutes les ressources portant la même balise.
10. Choisissez Suivant.
11. Sur la page Ajouter une stratégie d'accès, si le type que vous avez choisi était Standard et que vous souhaitez qu'Amazon EKS autorise le principal IAM à avoir des autorisations sur les objets Kubernetes de votre cluster, effectuez les étapes suivantes. Sinon, choisissez Next (Suivant).

    a. Pour Nom de la politique, choisissez une stratégie d'accès. Vous ne pouvez pas consulter les autorisations des stratégies d'accès, mais celles-ci incluent des

autorisations similaires à celles des objets `ClusterRole` destinés aux utilisateurs Kubernetes. Pour plus d'informations, consultez la section Rôles destinés aux utilisateurs dans la documentation Kubernetes.

b. Choisissez l'une des options suivantes :

- Cluster : choisissez cette option si vous souhaitez qu'Amazon EKS autorise le principal IAM à disposer des autorisations définies dans la stratégie d'accès pour tous les objets Kubernetes de votre cluster.
- Espace de noms Kubernetes : choisissez cette option si vous souhaitez qu'Amazon EKS autorise le principal IAM à disposer des autorisations définies dans la stratégie d'accès pour tous les objets Kubernetes dans un espace de nom Kubernetes spécifique de votre cluster. Pour Espace de noms, entrez le nom de l'espace de noms Kubernetes de votre cluster. Si vous souhaitez ajouter des espaces de noms supplémentaires, choisissez Ajouter un nouvel espace de noms et entrez son nom.

c. Pour ajouter des politiques supplémentaires, Sélectionnez Ajouter une politique. Vous pouvez définir la portée de chaque politique différemment, mais vous ne pouvez ajouter chaque politique qu'une seule fois.

d. Choisissez Suivant.

12. Vérifiez la configuration de votre entrée d'accès. Si quelque chose semble incorrect, choisissez Précédent pour revenir en arrière et corriger l'erreur. Si la configuration est correcte, choisissez Créer.

AWS CLI

Prérequis

La dernière version de la AWS CLI v1 installée et configurée sur votre appareil ou AWS CloudShell. AWS CLI La v2 ne prend pas en charge les nouvelles fonctionnalités depuis quelques jours. Vous pouvez vérifier votre version actuelle avec `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, reportez-vous à la section Installation, mise à jour et désinstallation de la AWS CLI configuration rapide `aws configure` dans le guide de l' AWS Command Line Interface utilisateur. La AWS CLI version installée dans le AWS CloudShell peut également être en retard de plusieurs versions par rapport à la dernière version. Pour le mettre à jour, consultez

la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

Pour créer une entrée d'accès

Vous pouvez utiliser l'un des exemples suivants pour créer des entrées d'accès :

- Créez une entrée d'accès pour un groupe de nœuds Linux Amazon EC2 autogéré. Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par votre Compte AWS identifiant et *eks-my-cluster-self-managed-NG-1* par le nom du rôle IAM de votre nœud. Si votre groupe de nœuds est un groupe de nœuds Windows, remplacez *EC2_Linux* par **EC2_Windows**.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1 --type EC2_Linux
```

  Vous ne pouvez pas utiliser l'option `--kubernetes-groups` lorsque vous spécifiez un type autre que `STANDARD`. Vous ne pouvez pas associer de stratégie d'accès à cette entrée d'accès, car son type est une valeur autre que `STANDARD`.

- Créez une entrée d'accès qui autorise un rôle IAM qui n'est pas utilisé pour un groupe de nœuds autogérés Amazon EC2, avec lequel vous souhaitez que Kubernetes autorise l'accès à votre cluster. Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par votre Compte AWS ID et *my-role par le nom de votre rôle* IAM. Remplacez *Utilisateurs* par le nom d'un groupe que vous avez spécifié dans un objet Kubernetes `RoleBinding` ou `ClusterRoleBinding` de votre cluster.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role --type STANDARD --user Viewers --
kubernetes-groups Viewers
```

- Créez une entrée d'accès qui permet à un utilisateur IAM de s'authentifier auprès de votre cluster. Cet exemple est fourni car c'est possible, bien que les bonnes pratiques IAM recommandent d'accéder à votre cluster en utilisant des rôles IAM dotés d'informations d'identification à court terme, plutôt que des utilisateurs IAM dotés d'informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le guide de l'utilisateur IAM.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:user/my-user --type STANDARD --username my-user
```

Si vous souhaitez que cet utilisateur dispose d'un accès à votre cluster supérieur aux autorisations associées aux rôles de découverte d'API Kubernetes, vous devez associer une stratégie d'accès à l'entrée d'accès, car l'option `--kubernetes-groups` n'est pas utilisée. Pour plus d'informations, consultez Associer les stratégies d'accès aux entrées d'accès et les dissocier des entrées d'accès et les rôles de découverte d'API dans la documentation Kubernetes.

Mise à jour des entrées d'accès

Vous pouvez mettre à jour une entrée d'accès à l'aide du AWS Management Console ou du AWS CLI.

AWS Management Console

Pour mettre à jour une entrée d'accès

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster pour lequel vous souhaitez créer une entrée d'accès.
3. Choisissez l'onglet Access.
4. Choisissez l'entrée d'accès que vous voulez mettre à jour.
5. Choisissez Modifier.
6. Pour Nom d'utilisateur, vous pouvez modifier la valeur existante.
7. Pour Groupes, vous pouvez supprimer les noms de groupes existants ou en ajouter de nouveaux. Si les noms de groupes suivants existent, ne les supprimez pas : system:nodes ou system:bootstrappers. La suppression de ces groupes peut entraîner un dysfonctionnement de votre cluster. Si vous ne spécifiez aucun nom de groupe et que vous souhaitez utiliser l'autorisation Amazon EKS, associez une stratégie d'accès ultérieurement.
8. Pour Balises, vous pouvez attribuer des étiquettes à l'entrée d'accès. Par exemple, pour faciliter la recherche de toutes les ressources portant la même balise. Vous pouvez également supprimer des balises existantes.
9. Sélectionnez Enregistrer les modifications.

10. Si vous souhaitez associer une stratégie d'accès à l'entrée, consultez Associer les stratégies d'accès aux entrées d'accès et les dissocier des entrées d'accès.

AWS CLI

Prérequis

Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

Pour mettre à jour une entrée d'accès

Remplacez *`my-cluster`* par le nom de votre cluster, *`111122223333`* par votre Compte AWS ID et *`EKS-my-cluster-my-namespace-Viewers`* par le nom d'un rôle IAM.

```
aws eks update-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --kubernetes-
groups Viewers
```

Vous ne pouvez pas utiliser l'option `--kubernetes-groups` si le type de l'entrée d'accès est une valeur autre que `STANDARD`. Vous ne pouvez pas non plus associer une stratégie d'accès à une entrée d'un type autre que `STANDARD`.

Suppression d'entrées d'accès

Si vous découvrez que vous avez supprimé une entrée d'accès par erreur, vous pouvez toujours la recréer. Si l'entrée d'accès que vous supprimez est associée à des stratégies d'accès, les associations sont automatiquement supprimées. Il n'est pas nécessaire de dissocier les stratégies d'accès d'une entrée d'accès avant de supprimer cette entrée d'accès.

Vous pouvez supprimer une entrée d'accès à l'aide du AWS Management Console ou du AWS CLI.

AWS Management Console

Pour supprimer une entrée d'accès

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster dont vous souhaitez supprimer une entrée d'accès.
3. Choisissez l'onglet Access.
4. Dans la liste des entrées d'accès, choisissez l'entrée d'accès que vous souhaitez supprimer.
5. Sélectionnez Delete (Supprimer).
6. Dans la boîte de dialogue de confirmation, choisissez Delete (Supprimer).

AWS CLI

Prérequis

Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

Pour supprimer une entrée d'accès

Remplacez *`my-cluster`* par le nom de votre cluster, *`111122223333`* par votre Compte AWS ID et *`my-role par le nom du rôle`* IAM auquel vous ne souhaitez plus avoir accès à votre cluster.

```
aws eks delete-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Associer les stratégies d'accès aux entrées d'accès et les dissocier des entrées d'accès

Vous pouvez attribuer une ou plusieurs stratégies d'accès à des entrées d'accès de type `STANDARD`. Amazon EKS accorde automatiquement aux autres types d'entrées d'accès les autorisations nécessaires pour fonctionner correctement dans votre cluster. Les stratégies d'accès d'Amazon EKS incluent des autorisations Kubernetes, et non des autorisations IAM. Avant d'associer une stratégie d'accès à une entrée d'accès, assurez-vous que vous connaissez bien les autorisations Kubernetes incluses dans chaque stratégie d'accès. Pour plus d'informations, consultez Autorisations des stratégies d'accès. Si aucune des stratégies d'accès ne répond à vos exigences, n'associez aucune stratégie d'accès à une entrée d'accès. Spécifiez plutôt un ou plusieurs noms de groupe pour l'entrée d'accès et créez et gérez des objets de contrôle d'accès basés sur des rôles Kubernetes. Pour plus d'informations, consultez Création d'entrées d'accès.

Prérequis

- Une entrée d'accès existante. Pour en créer un, consultez Création d'entrées d'accès.
- Un AWS Identity and Access Management rôle ou un utilisateur disposant des autorisations suivantes : `ListAccessEntries`,`DescribeAccessEntry`,`UpdateAccessEntry`,`ListAccessPolicies`,`AssociateAccessPolicy`et`DisassociateAccessPolicy`. Pour plus d'informations, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service dans la Référence des autorisations de service.

Avant d'associer des stratégies d'accès à des entrées d'accès, tenez compte des exigences suivantes :

- Vous pouvez associer plusieurs stratégies d'accès à chaque entrée d'accès, mais vous ne pouvez associer chaque politique à une entrée d'accès qu'une seule fois. Si vous associez plusieurs stratégies d'accès, le principal IAM de l'entrée d'accès dispose de toutes les autorisations incluses dans toutes les stratégies d'accès associées.
- Vous pouvez étendre une stratégie d'accès à toutes les ressources d'un cluster ou en spécifiant le nom d'un ou de plusieurs espaces de noms Kubernetes. Vous pouvez utiliser des caractères génériques pour le nom d'un espace de noms. Par exemple, si vous souhaitez étendre une stratégie d'accès à tous les espaces de noms commençant par `dev-`, vous pouvez spécifier `dev-*` sous forme de nom d'espace de noms. Assurez-vous que les espaces de noms existent sur votre cluster et que votre orthographe correspond au nom réel de l'espace de noms du cluster. Amazon EKS ne confirme ni l'orthographe ni l'existence des espaces de noms de votre cluster.

- Vous pouvez modifier la portée d'accès d'une stratégie d'accès après l'avoir associée à une entrée d'accès. Si vous avez défini la stratégie d'accès aux espaces de noms Kubernetes, vous pouvez ajouter et supprimer des espaces de noms pour l'association, si nécessaire.
- Si vous associez une stratégie d'accès à une entrée d'accès dont les noms de groupe sont également spécifiés, le principal IAM dispose alors de toutes les autorisations dans toutes les stratégies d'accès associées. Il dispose également de toutes les autorisations dans n'importe quel objet Kubernetes `Role` ou `ClusterRole` spécifié dans n'importe quels objets Kubernetes `Role` et `RoleBinding` qui spécifient les noms de groupes.
- Si vous exécutez la commande `kubectl auth can-i --list`, vous ne verrez aucune autorisation Kubernetes attribuée par les stratégies d'accès associées à une entrée d'accès pour le principal IAM que vous utilisez lorsque vous exécutez la commande. La commande affiche les autorisations Kubernetes uniquement si vous les avez accordées dans les objets Kubernetes `Role` ou `ClusterRole` que vous avez liés aux noms de groupe ou au nom d'utilisateur que vous avez spécifiés pour une entrée d'accès.
- Si vous vous faites passer pour un utilisateur ou un groupe Kubernetes lorsque vous interagissez avec des objets Kubernetes de votre cluster, par exemple en utilisant la commande `kubectl` avec **`--as`** ***`username`*** ou **`--as-group`** ***`group-name`***, vous forcez l'utilisation de l'autorisation Kubernetes RBAC. Par conséquent, le principal IAM ne dispose d'aucune autorisation attribuée par les stratégies d'accès associées à l'entrée d'accès. Les seules autorisations Kubernetes dont dispose l'utilisateur ou le groupe, pour lequel le principal IAM se fait passer, sont les autorisations Kubernetes que vous lui avez accordées dans les objets Kubernetes `Role` ou `ClusterRole` que vous avez liés aux noms de groupe ou au nom d'utilisateur. Pour que votre principal IAM dispose des autorisations définies dans les stratégies d'accès associées, ne vous faites pas passer pour un utilisateur ou un groupe Kubernetes. Le principal IAM disposera également encore de toutes les autorisations que vous lui avez octroyées dans les objets Kubernetes `Role` ou `ClusterRole` que vous avez liés aux noms de groupe ou au nom d'utilisateur que vous avez spécifiés pour l'entrée d'accès. Pour plus d'informations, consultez Emprunt de l'identité de l'utilisateur dans la documentation Kubernetes.

Vous pouvez associer une politique d'accès à une entrée d'accès à l'aide du AWS Management Console ou du AWS CLI.

AWS Management Console

Pour associer une stratégie d'accès à une entrée d'accès à l'aide de la AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster qui possède une entrée d'accès à laquelle associer une stratégie d'accès.
3. Choisissez l'onglet Access.
4. Si le type d'entrée d'accès est Standard, vous pouvez associer ou dissocier les stratégies d'accès Amazon EKS. Si le type de votre entrée d'accès est autre que Standard, cette option n'est pas disponible.
5. Choisissez Associer la stratégie d'accès.
6. Dans Nom de la stratégie, sélectionnez la stratégie avec les autorisations que vous souhaitez attribuer au principal IAM. Pour consulter les autorisations incluses dans chaque stratégie, consultez Autorisations des stratégies d'accès.
7. Pour Portée d'accès, choisissez une portée d'accès. Si vous choisissez Cluster, les autorisations définies dans la stratégie d'accès sont accordées au principal IAM pour les ressources de tous les espaces de noms Kubernetes. Si vous choisissez un espace de noms Kubernetes, vous pouvez ensuite choisir Ajouter un espace de noms. Dans le champ Espace de noms qui apparaît, vous pouvez saisir le nom d'un espace de noms Kubernetes sur votre cluster. Si vous souhaitez que le principal IAM dispose d'autorisations sur plusieurs espaces de noms, vous pouvez saisir plusieurs espaces de noms.
8. Choisissez Ajouter une stratégie d'accès.

AWS CLI

Prérequis

Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec

aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

Pour associer une stratégie d'accès à une entrée d'accès

1. Affichez les stratégies d'accès disponibles.

```
aws eks list-access-policies --output table
```

L'exemple qui suit illustre un résultat.

```
-------------------------------------------------------------------------------------
|                                        ListAccessPolicies                         |
                       |
+-----------------------------------------------------------------------------------
+
||                                          accessPolicies                           |
                      ||
|+-------------------------------------------------------------------
+-------------------------------+|
||                                 arn                                |
     name              ||
|+-------------------------------------------------------------------
+-------------------------------+|
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy       |
 AmazonEKSAdminPolicy         ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy |
 AmazonEKSClusterAdminPolicy  ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy        |
 AmazonEKSEditPolicy          ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy        |
 AmazonEKSViewPolicy          ||
|+-------------------------------------------------------------------
+-------------------------------+|
```

Pour consulter les autorisations incluses dans chaque stratégie, consultez Autorisations des stratégies d'accès.

2. Consultez vos entrées d'accès existantes. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks list-access-entries --cluster-name my-cluster
```

L'exemple qui suit illustre un résultat.

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/my-role",
        "arn:aws:iam::111122223333:user/my-user"
    ]
}
```

3. Associez une stratégie d'accès à une entrée d'accès. L'exemple suivant associe la stratégie d'accès `AmazonEKSViewPolicy` à une entrée d'accès. Chaque fois que le rôle IAM *my-role* tente d'accéder aux objets Kubernetes du cluster, Amazon EKS autorise le rôle à utiliser les autorisations définies dans la stratégie pour accéder aux objets Kubernetes des espaces de noms Kubernetes *my-namespace1* et *my-namespace2* uniquement. Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par l'identifiant de votre Compte AWS et *my-role* par le nom du rôle IAM pour lequel vous souhaitez qu'Amazon EKS autorise l'accès aux objets du cluster Kubernetes.

```
aws eks associate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --access-scope type=namespace,namespaces=my-namespace1,my-namespace2 --
policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy
```

Si vous souhaitez que le principal IAM dispose des autorisations à l'échelle du cluster, remplacez **`type=namespace,namespaces=`*`my-namespace1`*`,`*`my-namespace2`* par **`type=cluster`**. Si vous souhaitez associer plusieurs stratégies d'accès à l'entrée d'accès, exécutez la commande plusieurs fois, chaque fois avec une stratégie d'accès unique. Chaque stratégie d'accès associée possède sa propre portée.

> **Note**
>
> Si vous souhaitez ultérieurement modifier la portée d'une stratégie d'accès associée, réexécutez la commande précédente avec la nouvelle portée. Par exemple, si vous souhaitez supprimer *my-namespace2*, vous devez réexécuter la commande en utilisant uniquement **`type=namespace,namespaces=`*`my-namespace1`***. Si vous souhaitez modifier la portée de **`namespace`** à **`cluster`**, vous devez

exécuter à nouveau la commande en utilisant **type=cluster**, ce qui supprime **type=namespace,namespaces=*my-namespace1*,*my-namespace2***.

Pour dissocier une stratégie d'accès d'une entrée d'accès

1. Déterminez les stratégies d'accès associées à une entrée d'accès.

```
aws eks list-associated-access-policies --cluster-name my-cluster --principal-
arn arn:aws:iam::111122223333:role/my-role
```

L'exemple qui suit illustre un résultat.

```
{
    "clusterName": "my-cluster",
    "principalArn": "arn:aws:iam::111122223333",
    "associatedAccessPolicies": [
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSViewPolicy",
            "accessScope": {
                "type": "cluster",
                "namespaces": []
            },
            "associatedAt": "2023-04-17T15:25:21.675000-04:00",
            "modifiedAt": "2023-04-17T15:25:21.675000-04:00"
        },
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSAdminPolicy",
            "accessScope": {
                "type": "namespace",
                "namespaces": [
                    "my-namespace1",
                    "my-namespace2"
                ]
            },
            "associatedAt": "2023-04-17T15:02:06.511000-04:00",
            "modifiedAt": "2023-04-17T15:02:06.511000-04:00"
        }
    ]
```

```
}
```

Dans l'exemple précédent, le principal IAM pour cette entrée d'accès dispose d'autorisations d'affichage sur tous les espaces de noms du cluster et d'autorisations d'administrateur pour deux espaces de noms Kubernetes.

2. Dissociez une stratégie d'accès d'une entrée d'accès. Dans cet exemple, la stratégie `AmazonEKSAdminPolicy` est dissociée d'une entrée d'accès. Le principal IAM conserve toutefois les autorisations définies dans la stratégie d'accès `AmazonEKSViewPolicy` pour les objets des espaces de noms *`my-namespace1`* et *`my-namespace2`*, car cette stratégie d'accès n'est pas dissociée de l'entrée d'accès.

```
aws eks disassociate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy
```

Autorisations des stratégies d'accès

Les stratégies d'accès incluent `rules` qui contiennent Kubernetes `verbs` (autorisations) et `resources`. Les stratégies d'accès n'incluent pas les autorisations ou les ressources IAM. Similaires aux objets Kubernetes `Role` et`ClusterRole`, les stratégies d'accès incluent uniquement `allow` `rules`. Vous ne pouvez pas modifier le contenu d'une stratégie d'accès. Vous ne pouvez pas créer vos propres stratégies d'accès. Si les autorisations définies dans les stratégies d'accès ne répondent pas à vos besoins, créez alors des objets Kubernetes RBAC et spécifiez les noms de groupe pour vos entrées d'accès. Pour plus d'informations, consultez Création d'entrées d'accès. Les autorisations contenues dans les stratégies d'accès sont similaires à celles des rôles de cluster destinés aux utilisateurs Kubernetes. Pour plus d'informations, consultez la section Rôles destinés aux utilisateurs dans la documentation Kubernetes.

Choisissez n'importe quelle stratégie d'accès pour voir son contenu. Chaque ligne de chaque table dans chaque stratégie d'accès est une règle distincte.

Amazon Eks AdminPolicy

Cette stratégie d'accès inclut des autorisations qui accordent à un principal IAM le plus grand nombre d'autorisations sur les ressources. Lorsqu'elle est associée à une entrée d'accès, sa portée d'accès est généralement constituée d'un ou de plusieurs espaces de noms Kubernetes. Si vous souhaitez qu'un principal IAM dispose d'un accès administrateur à toutes les ressources de votre cluster, associez plutôt la stratégie d'accès Amazon Eks ClusterAdminPolicy à votre entrée d'accès.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy`

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
apps	daemonsets , deployments , deployments/rollback , deployments/scale , replicasets , replicasets/scale, statefulsets , statefulsets/scale	create, delete, deletecollection , patch, update
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
authorization.k8s.io	localsubjectaccessreviews	create
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
batch	cronjobs, jobs	create, delete, deletecollection , patch, update

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
rbac.authorization.k8s.io	rolebindings , roles	create, delete, deletecollection , get, list, patch, update, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get,list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	pods, pods/attach , pods/exec , pods/portforward , pods/proxy	create, delete, deletecollection , patch, update
	serviceaccounts	impersonate
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch
	namespaces	get,list, watch

Amazon Eks ClusterAdminPolicy

Cette stratégie d'accès inclut des autorisations qui accordent à un administrateur principal IAM l'accès à un cluster. Lorsqu'elle est associée à une entrée d'accès, sa portée d'accès est généralement le cluster, plutôt qu'un espace de noms Kubernetes. Si vous souhaitez qu'un principal IAM ait une portée administrative plus limitée, pensez plutôt à associer la stratégie d'accès Amazon Eks AdminPolicy à votre entrée d'accès.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy`

Groupes d'API Kubernetes	Kubernetes nonResourceURLs	Ressources Kubernetes	Verbes (autorisations) Kubernetes
*		*	*
	*		*

Amazon Eks AdminViewPolicy

Cette politique d'accès inclut des autorisations qui accordent à un IAM principal l'accès à la liste/à l'affichage de toutes les ressources d'un cluster. Notez que cela inclut Kubernetesles secrets.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy`

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
*	*	`get, list, watch`

Amazon Eks EditPolicy

Cette stratégie d'accès inclut des autorisations qui permettent au principal IAM de modifier la plupart des ressources Kubernetes.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy`

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
`apps`	`daemonsets` , `deployments` , `deployments/rollback` , `deployments/scale` , `replicasets` , `replicasets/`	`create`, `delete`, `deletecollection` , `patch`, `update`

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
	scale, statefulsets , statefulsets/scale	
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
	namespaces	get, list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	serviceaccounts	impersonate
	pods, pods/attach , pods/exec , pods/portforward , pods/proxy	create, delete, deletecollection , patch, update
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get, list, watch

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
	`bindings`, `events`, `limitranges` , `namespaces/status` , `pods/log`, `pods/status` , `replicationcontrollers/status` , `resourcequotas` , `resourcequotas/status`	`get`, `list`, `watch`

Amazon Eks ViewPolicy

Cette stratégie d'accès inclut des autorisations qui permettent à un principal IAM de consulter la plupart des ressources Kubernetes.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy`

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
`apps`	`controllerrevisions` , `daemonsets` , `daemonsets/status` , `deployments` , `deployments/scale` , `deployments/status` , `replicasets` , `replicasets/scale` , `replicasets/status` , `statefulsets` , `statefulsets/scale` , `statefulsets/status`	`get`, `list`, `watch`
`autoscaling`	`horizontalpodautoscalers` , `horizonta`	`get`, `list`, `watch`

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
	lpodautoscalers/status	
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch

Groupes d'API Kubernetes	Ressources Kubernetes	Verbes (autorisations) Kubernetes
	`configmaps` , `endpoints` , `persistentvolumeclaims` , `persistentvolumeclaims/status` , `pods`, `replicationcontrollers` , `replicationcontrollers/scale` , `serviceaccounts` , `services`, `services/status`	`get`, `list`, `watch`
	`bindings`, `events`, `limitranges` , `namespaces/status` , `pods/log`, `pods/status` , `replicationcontrollers/status` , `resourcequotas` , `resourcequotas/status`	`get`, `list`, `watch`
	`namespaces`	`get`, `list`, `watch`

Mises à jour des stratégies d'accès

Affichez les détails sur les mises à jour des stratégies d'accès depuis leur introduction. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS dans la page de l'historique des documents Amazon EKS.

Modification	Description	Date
Addition `AmazonEKSAdminViewPolicy`	Ajoutez une nouvelle politique pour un accès étendu aux vues, y compris à des ressources telles que les secrets.	23 avril 2024

Modification	Description	Date
Stratégies d'accès introduites.	Amazon EKS a introduit des stratégies d'accès.	29 mai 2023

Migration des entrées `aws-auth ConfigMap` existantes pour accéder aux entrées

Si vous avez ajouté des entrées de `ConfigMap` `aws-auth` à votre cluster, nous vous recommandons de créer des entrées d'accès pour les entrées existantes dans votre cluster `ConfigMap` `aws-auth`. Après avoir créé les entrées d'accès, vous pouvez les supprimer de votre `ConfigMap`. Vous ne pouvez pas associer de stratégies d'accès aux entrées dans la `ConfigMap` `aws-auth`. Si vous souhaitez associer des stratégies d'accès à vos principaux IAM, créez des entrées d'accès.

> **Important**
>
> Ne supprimez pas les entrées existantes de `ConfigMap` `aws-auth` créées par Amazon EKS lorsque vous avez ajouté un groupe de nœuds gérés ou un profil Fargate à votre cluster. Si vous supprimez les entrées créées par Amazon EKS dans la `ConfigMap`, votre cluster ne fonctionnera pas correctement. Vous pouvez toutefois supprimer toutes les entrées des groupes de nœuds autogérés après avoir créé des entrées d'accès pour ceux-ci.

Prérequis

- Avoir des connaissances sur les entrées d'accès et les stratégies d'accès. Pour plus d'informations, consultez Gérer les entrées d'accès et Associer les stratégies d'accès aux entrées d'accès et les dissocier des entrées d'accès.
- Un cluster existant dont la version de plateforme correspond ou est ultérieure aux versions répertoriées dans les prérequis de la rubrique Autoriser les rôles IAM ou les utilisateurs IAM à accéder aux objets Kubernetes de votre cluster Amazon EKS.
- Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

- Autorisations Kubernetes permettant de modifier la `ConfigMap` `aws-auth` dans l'espace de noms `kube-system`.
- Un AWS Identity and Access Management rôle ou un utilisateur disposant des autorisations suivantes : `CreateAccessEntry` et`ListAccessEntries`. Pour plus d'informations, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service dans la Référence des autorisations de service.

Pour migrer une entrée de votre **`aws-auth ConfigMap`** vers une entrée d'accès

1. Consultez les entrées existantes dans votre `aws-auth ConfigMap`. Remplacez *`my-cluster`* par le nom de votre cluster.

   ```
   eksctl get iamidentitymapping --cluster my-cluster
   ```

   L'exemple qui suit illustre un résultat.

   ```
   ARN
                USERNAME                                GROUPS
                            ACCOUNT
   arn:aws:iam::111122223333:role/EKS-my-cluster-Admins
                Admins                                  system:masters
   arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers
                my-namespace-Viewers                    Viewers
   arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1
                               system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   arn:aws:iam::111122223333:user/my-user
                my-user
   arn:aws:iam::111122223333:role/EKS-my-cluster-fargateprofile1
                               system:node:{{SessionName}}
    system:bootstrappers,system:nodes,system:node-proxier
   arn:aws:iam::111122223333:role/EKS-my-cluster-managed-ng
                               system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   ```

2. Créez des entrées d'accès pour toutes les entrées `ConfigMap` que vous avez créées et renvoyées dans le résultat précédent. Lorsque vous créez les entrées d'accès, assurez-vous de spécifier les mêmes valeurs pour `ARN`, `USERNAME`, `GROUPS` et `ACCOUNT` et de les renvoyer dans votre résultat. Dans l'exemple de résultat, vous devez créer des entrées d'accès pour toutes les

entrées sauf les deux dernières, étant donné que ces entrées ont été créées par Amazon EKS pour un profil Fargate et un groupe de nœuds géré.

3. Supprimez les entrées de la `ConfigMap` pour toutes les entrées d'accès que vous avez créées. Si vous ne supprimez pas l'entrée de la `ConfigMap`, les paramètres de l'entrée d'accès pour l'ARN principal IAM remplaceront l'entrée `ConfigMap`. Remplacez *`111122223333`* par votre Compte AWS identifiant et *`EKS-my-cluster-my-namespace-Viewers`* par le nom du rôle figurant dans l'entrée de votre. `ConfigMap` Si l'entrée que vous supprimez concerne un utilisateur IAM plutôt qu'un rôle IAM, remplacez **`role`** par **`user`** et *`EKS-my-cluster-my-namespace-viewers`* par le nom d'utilisateur.

```
eksctl delete iamidentitymapping --arn arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --cluster my-cluster
```

Autorisation d'un principal IAM à accéder à votre cluster

> **Important**
>
> Le `aws-auth` ConfigMap est obsolète. La méthode recommandée pour gérer l'accès aux Kubernetes API est Access Entries.

L'authentificateur IAM AWS pour Kubernetes, qui s'exécute sur le plan de contrôle Amazon EKS, permet à des principaux IAM d'accéder à votre cluster. L'authentificateur obtient ses informations de configuration à partir du `aws-auth` `ConfigMap`. Pour tous les paramètres `aws-auth` `ConfigMap`, consultez Format de configuration complet sur GitHub.

Ajout de principaux IAM à votre cluster Amazon EKS

Lorsque vous créez un cluster Amazon EKS, le principal IAM qui crée le cluster se voit automatiquement accorder des autorisations `system:masters` dans la configuration du contrôle d'accès basé sur les rôles (RBAC) du cluster dans le plan de contrôle Amazon EKS. Ce principal n'apparaît dans aucune configuration visible. Souvenez-vous donc du principal qui a créé le cluster à l'origine. Pour permettre à d'autres principaux IAM d'interagir avec votre cluster, modifiez la `ConfigMap` `aws-auth` dans Kubernetes et créez un `rolebinding` ou un `clusterrolebinding` Kubernetes avec le nom d'un `group` que vous spécifiez dans la `ConfigMap` `aws-auth`.

> **Note**
>
> Pour plus d'informations sur la configuration du contrôle d'accès basé sur les rôles (RBAC) de Kubernetes, consultez la rubrique Utilisation de l'autorisation RBAC de la documentation Kubernetes.

Pour ajouter un principal IAM à un cluster Amazon EKS

1. Déterminer les informations d'identification que `kubectl` utilise pour accéder à votre cluster. Sur votre ordinateur, vous pouvez voir quelles informations d'identification `kubectl` utilise avec la commande suivante. Remplacez ***~/.kube/config*** par le chemin de votre fichier `kubeconfig` si vous n'utilisez pas le chemin par défaut.

   ```
   cat ~/.kube/config
   ```

   L'exemple qui suit illustre un résultat.

   ```
   [...]
   contexts:
   - context:
       cluster: my-cluster.region-code.eksctl.io
       user: admin@my-cluster.region-code.eksctl.io
     name: admin@my-cluster.region-code.eksctl.io
   current-context: admin@my-cluster.region-code.eksctl.io
   [...]
   ```

   Dans l'exemple de sortie précédent, les informations d'identification d'un utilisateur nommé *`admin`* sont configurées pour un cluster nommé *`my-cluster`*. Si c'est l'utilisateur qui a créé le cluster, alors il a déjà accès à votre cluster. S'il ne s'agit pas de l'utilisateur qui a créé le cluster, suivez les étapes restantes pour permettre à d'autres principaux IAM d'accéder au cluster. Les bonnes pratiques IAM recommandent d'accorder des autorisations à des rôles plutôt qu'à des utilisateurs. Vous pouvez voir quels autres principaux ont actuellement accès à votre cluster à l'aide de la commande suivante :

   ```
   kubectl describe -n kube-system configmap/aws-auth
   ```

   L'exemple qui suit illustre un résultat.

```
Name:         aws-auth
Namespace:    kube-system
Labels:       <none>
Annotations:  <none>

Data
====
mapRoles:
----
- groups:
  - system:bootstrappers
  - system:nodes
  rolearn: arn:aws:iam::111122223333:role/my-node-role
  username: system:node:{{EC2PrivateDNSName}}


BinaryData
====

Events:  <none>
```

L'exemple précédent est un `aws-auth` `ConfigMap` par défaut. Seul le rôle d'instance du nœud a accès au cluster.

2. Assurez-vous de disposer de `roles` et de `rolebindings` ou de `clusterroles` et de `clusterrolebindings` Kubernetes auxquels vous pouvez mapper des principaux IAM. Pour plus d'informations sur ces ressources, consultez Utilisation de l'autorisation RBAC dans la documentation Kubernetes.

   1. Affichez vos Kubernetes `roles` ou `clusterroles` existants. Les `Roles` sont étendus à un `namespace`, mais les `clusterroles` sont étendus au cluster.

      ```
      kubectl get roles -A
      ```

      ```
      kubectl get clusterroles
      ```

   2. Affichez les détails de tout `role` ou `clusterrole` renvoyé dans la sortie précédente et assurez-vous qu'il dispose des autorisations (`rules`) dont vous souhaitez que vos principaux IAM disposent dans votre cluster.

Remplacez *role-name* par un nom de `role` renvoyé dans la sortie de la commande précédente. Remplacez *kube-system* par l'espace de noms du `role`.

```
kubectl describe role role-name -n kube-system
```

Remplacez *cluster-role-name* par un nom de `clusterrole` renvoyé dans la sortie de la commande précédente.

```
kubectl describe clusterrole cluster-role-name
```

3. Affichez vos Kubernetes `rolebindings` ou `clusterrolebindings` existants. Les `Rolebindings` sont étendus à un `namespace`, mais les `clusterrolebindings` sont étendus au cluster.

```
kubectl get rolebindings -A
```

```
kubectl get clusterrolebindings
```

4. Affichez les détails de n'importe quel `rolebinding` ou `clusterrolebinding` et confirmez qu'il possède un `role` ou un `clusterrole` de l'étape précédente répertorié comme un `roleRef` et un nom de groupe répertorié pour `subjects`.

   Remplacez *role-binding-name* par un nom de `rolebinding` renvoyé dans la sortie de la commande précédente. Remplacez *kube-system* avec le `namespace` du `rolebinding`.

```
kubectl describe rolebinding role-binding-name -n kube-system
```

   L'exemple qui suit illustre un résultat.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eks-console-dashboard-restricted-access-role-binding
  namespace: default
subjects:
- kind: Group
  name: eks-console-dashboard-restricted-access-group
  apiGroup: rbac.authorization.k8s.io
```

```
roleRef:
  kind: Role
  name: eks-console-dashboard-restricted-access-role
  apiGroup: rbac.authorization.k8s.io
```

Remplacez *cluster-role-binding-name* par un nom de `clusterrolebinding` renvoyé dans la sortie de la commande précédente.

```
kubectl describe clusterrolebinding cluster-role-binding-name
```

L'exemple qui suit illustre un résultat.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks-console-dashboard-full-access-binding
subjects:
- kind: Group
  name: eks-console-dashboard-full-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: eks-console-dashboard-full-access-clusterrole
  apiGroup: rbac.authorization.k8s.io
```

3. Modifiez le `aws-auth ConfigMap`. Vous pouvez utiliser un outil tel que `eksctl` pour mettre à jour le `ConfigMap` ou vous pouvez le mettre à jour manuellement en le modifiant.

> **Important**
>
> Nous vous recommandons d'utiliser `eksctl`, ou un autre outil, pour modifier le `ConfigMap`. Pour plus d'informations sur les autres outils que vous pouvez utiliser, consultez Utiliser des outils pour apporter des modifications au `aws-authConfigMap` dans les guides des bonnes pratiques Amazon EKS. Un `aws-auth ConfigMap` mis en forme incorrectement peut entraîner la perte de l'accès à votre cluster.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

1. Affichez les mappages actuels dans le `ConfigMap`. Remplacez *`my-cluster`* par le nom de votre cluster. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

   ```
   eksctl get iamidentitymapping --cluster my-cluster --region=region-code
   ```

   L'exemple qui suit illustre un résultat.

   ```
   ARN
                       USERNAME                                GROUPS
                ACCOUNT
   arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
   NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   ```

2. Ajoutez un mappage pour un rôle. Remplacez *`my-role`* par le nom de votre rôle. Remplacez *`eks-console-dashboard-full-access-group`* par le nom du groupe spécifié dans votre Kubernetes `RoleBinding` ou objet `ClusterRoleBinding`. Remplacez *`111122223333`* par votre ID de compte. Vous pouvez remplacer *`admin`* par n'importe quel nom que vous choisissez.

   ```
   eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
       --arn arn:aws:iam::111122223333:role/my-role --username admin --group eks-
   console-dashboard-full-access-group \
       --no-duplicate-arns
   ```

   > **Important**
   >
   > L'ARN de rôle ne peut pas inclure de chemin d'accès tel que `role/my-team/developers/my-role`. Le format de l'ARN doit être `arn:aws:iam::111122223333:role/my-role`. Dans cet exemple, `my-team/developers/` doit être supprimé.

L'exemple qui suit illustre un résultat.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-
role" to auth ConfigMap
```

3. Ajoutez un mappage pour un utilisateur. Les bonnes pratiques IAM recommandent d'accorder des autorisations à des rôles plutôt qu'à des utilisateurs. Remplacez *my-user* par votre nom d'utilisateur. Remplacez *eks-console-dashboard-restricted-access-group* par le nom du groupe spécifié dans votre Kubernetes `RoleBinding` ou objet `ClusterRoleBinding`. Remplacez *111122223333* par votre ID de compte. Vous pouvez remplacer *my-user* par n'importe quel nom que vous choisissez.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user --username my-user --
group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

L'exemple qui suit illustre un résultat.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-
user" to auth ConfigMap
```

4. Affichez de nouveau les mappages dans le `ConfigMap`.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

L'exemple qui suit illustre un résultat.

```
ARN
                  USERNAME                                GROUPS
                    ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/admin
                      my-role                                 eks-console-
dashboard-full-access-group
```

```
arn:aws:iam::111122223333:user/my-user
                        my-user                                    eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

1. Ouvrez le `ConfigMap` pour le modifier.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   > **Note**
   >
   > Si vous recevez une erreur indiquant « `Error from server (NotFound): configmaps "aws-auth" not found` », suivez la procédure dans Appliquer la `ConfigMap aws-auth` à votre cluster pour appliquer le `ConfigMap` stock.

2. Ajoutez vos principaux IAM à la `ConfigMap`. Un groupe IAM n'est pas un principal IAM, il ne peut donc pas être ajouté à la `ConfigMap`.
   - Pour ajouter un rôle IAM (par exemple, pour les utilisateurs fédérés) : ajoutez les détails du rôle à la section `mapRoles` du `ConfigMap`, sous `data`. Ajoutez cette section si elle n'existe pas déjà dans le fichier. Chaque entrée prend en charge les paramètres suivants :
     - rolearn : ARN du rôle IAM à ajouter. Cette valeur ne peut pas inclure un chemin. Par exemple, vous ne pouvez pas spécifier un ARN tel que `arn:aws:iam::111122223333:role/my-team/developers/role-name`. L'ARN doit être `arn:aws:iam::111122223333:role/role-name`.
     - username : le nom d'utilisateur dans Kubernetes à mapper au rôle IAM.
     - groups : le groupe ou la liste des groupes Kubernetes auxquels mapper le rôle. Le groupe peut être un groupe par défaut ou un groupe spécifié dans un `clusterrolebinding` ou `rolebinding`. Pour plus d'informations, consultez Rôles par défaut et liaisons de rôles dans la documentation Kubernetes.
   - Pour ajouter un utilisateur IAM : les bonnes pratiques IAM recommandent d'accorder des autorisations à des rôles plutôt qu'à des utilisateurs. Ajoutez les détails de l'utilisateur dans la section `mapUsers` de la `ConfigMap`, sous `data`. Ajoutez cette section si elle n'existe pas déjà dans le fichier. Chaque entrée prend en charge les paramètres suivants :

- userarn : ARN de l'utilisateur IAM à ajouter.
- username : le nom d'utilisateur dans Kubernetes à mapper à l'utilisateur IAM.
- groups : le groupe ou la liste des groupes Kubernetes auxquels mapper l'utilisateur. Le groupe peut être un groupe par défaut ou un groupe spécifié dans un `clusterrolebinding` ou `rolebinding`. Pour plus d'informations, consultez Rôles par défaut et liaisons de rôles dans la documentation Kubernetes.

Par exemple, le bloc YAML ci-dessous contient :

- Une section `mapRoles` qui mappe l'instance du nœud IAM avec des groupes Kubernetes afin que les nœuds puissent s'inscrire eux-mêmes avec le cluster et le rôle IAM `my-console-viewer-role` mappé à un groupe Kubernetes pouvant afficher toutes les ressources Kubernetes pour tous les clusters. Pour obtenir la liste des autorisations de groupe IAM et Kubernetes requises pour le rôle IAM `my-console-viewer-role`, voir Autorisations nécessaires.
- `mapUsers`Section qui fait correspondre l'utilisateur `admin` IAM du AWS compte par défaut au `system:masters` Kubernetes groupe et l'`my-user`utilisateur d'un autre AWS compte mappé à un Kubernetes groupe qui peut afficher les Kubernetes ressources d'un espace de noms spécifique. Pour obtenir la liste des autorisations de groupe IAM et Kubernetes requises pour l'utilisateur IAM `my-user`, voir Autorisations nécessaires.

Ajoutez ou supprimez des lignes si nécessaire et remplacez toutes les *`example values`* par vos propres valeurs.

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this
 file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::111122223333:role/my-role
      username: system:node:{{EC2PrivateDNSName}}
    - groups:
      - eks-console-dashboard-full-access-group
      rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
```

```
      username: my-console-viewer-role
  mapUsers: |
    - groups:
      - system:masters
      userarn: arn:aws:iam::111122223333:user/admin
      username: admin
    - groups:
      - eks-console-dashboard-restricted-access-group
      userarn: arn:aws:iam::444455556666:user/my-user
      username: my-user
```

3. Enregistrez le fichier et quittez votre éditeur de texte.

Appliquer la **ConfigMap aws-auth** à votre cluster

Le `aws-auth` `ConfigMap` est automatiquement créé et appliqué à votre cluster lorsque vous créez un groupe de nœuds géré ou lorsque vous créez un groupe de nœuds à l'aide de `eksctl`. Il est initialement créé pour permettre aux nœuds de rejoindre votre cluster, mais vous utilisez également cette `ConfigMap` pour ajouter un accès RBAC aux principaux IAM. Si vous avez lancé des nœuds autogérés et que vous n'avez pas encore appliqué la `aws-auth` `ConfigMap` à votre cluster, vous pouvez le faire en suivant la procédure ci-dessous.

Pour appliquer le **aws-authConfigMap** à votre cluster

1. Vérifiez si vous avez déjà appliqué la `aws-auth` `ConfigMap`.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

   Si vous recevez une erreur indiquant « `Error from server (NotFound): configmaps "aws-auth" not found` », effectuez les opérations suivantes pour appliquer le `ConfigMap` stock.

2. Téléchargez, modifiez et appliquez le plan de configuration de l' AWS authentificateur.

   a. Téléchargez la mappe de configuration.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

b. Dans le fichier `aws-auth-cm.yaml`, remplacez `rolearn` par l'Amazon Resource Name (ARN) du rôle IAM associé par vos nœuds. Pour ce faire, utilisez un éditeur de texte ou remplacez *`my-node-instance-role`* et exécutez la commande suivante :

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
```

Ne modifiez aucune autre ligne de ce fichier.

> **Important**
>
> L'ARN de rôle ne peut pas inclure de chemin d'accès tel que `role/my-team/developers/my-role`. Le format de l'ARN doit être `arn:aws:iam::111122223333:role/my-role`. Dans cet exemple, `my-team/developers/` doit être supprimé.

Vous pouvez inspecter les sorties de la AWS CloudFormation pile pour vos groupes de nœuds et rechercher les valeurs suivantes :

- InstanceRoleARN — Pour les groupes de nœuds créés avec `eksctl`
- NodeInstanceRôle — Pour les groupes de nœuds créés à l'aide de AWS CloudFormation modèles Amazon EKS vendus dans le AWS Management Console

c. Appliquez la configuration. L'exécution de cette commande peut prendre quelques minutes.

```
kubectl apply -f aws-auth-cm.yaml
```

> **Note**
>
> Si vous recevez d'autres erreurs concernant les types d'autorisations ou de ressources, consultez Accès non autorisé ou refusé (`kubectl`) dans la rubrique relative à la résolution des problèmes.

3. Observez le statut de vos nœuds et attendez qu'ils obtiennent le statut `Ready`.

```
kubectl get nodes --watch
```

Saisissez `Ctrl+C` pour revenir à une invite de shell.

Authentifier les utilisateurs de votre cluster auprès d'un fournisseur d'OpenID Connectidentité

Amazon EKS prend en charge l'utilisation de fournisseurs d'identité OpenID Connect (OIDC) comme méthode pour authentifier les utilisateurs auprès de votre cluster. OIDCles fournisseurs d'identité peuvent être utilisés avec ou comme alternative à AWS Identity and Access Management (IAM). Pour plus d'informations sur l'utilisation d'IAM, consultez the section called "Autoriser l'accès aux API Kubernetes". Après avoir configuré l'authentification dans votre cluster, vous pouvez créer des Kubernetes `roles` et `clusterroles` pour attribuer des autorisations aux rôles, puis lier les rôles aux identités à l'aide de Kubernetes `rolebindings` et `clusterrolebindings`. Pour plus d'informations, consultez Utilisation de l'autorisation RBAC dans la documentation Kubernetes.

Considérations

- Vous pouvez associer un fournisseur d'identité OIDC à votre cluster.
- Kubernetes ne fournit pas de fournisseur d'identité OIDC. Vous pouvez utiliser un fournisseur d'identité OIDC public existant ou vous pouvez exécuter votre propre fournisseur d'identité. Pour obtenir la liste des fournisseurs certifiés, consultez Certification OpenID sur le site OpenID.
- L'URL de l'émetteur du fournisseur d'identité OIDC doit être accessible publiquement, afin qu'Amazon EKS puisse découvrir les clés de signature. Amazon EKS ne prend pas en charge les fournisseurs d'identité OIDC dotés de certificats auto-signés.
- Vous ne pouvez pas désactiver l'authentification IAM dans votre cluster, car elle est toujours nécessaire pour joindre des nœuds à un cluster.
- Un cluster Amazon EKS doit toujours être créé par un principal AWS IAM, plutôt que par un utilisateur du fournisseur d'OIDCidentité. En effet, le créateur de cluster interagit avec les API Amazon EKS et non pas avec les API Kubernetes.
- OIDCles utilisateurs authentifiés par le fournisseur d'identité sont répertoriés dans le journal d'audit du cluster si les CloudWatch journaux sont activés pour le plan de contrôle. Pour plus d'informations, consultez Activation et désactivation des journaux de plan de contrôle.
- Vous ne pouvez pas vous connecter au AWS Management Console avec un compte auprès d'un OIDC fournisseur. Vous ne pouvez consulter les Kubernetes ressources dans la console qu'en vous connectant à l' AWS Management Console aide d'un AWS Identity and Access Management compte.

Associer un fournisseur d'identité OIDC

Pour pouvoir associer un fournisseur d'identité OIDC à votre cluster, vous devez obtenir les informations suivantes de votre fournisseur :

URL de l'émetteur

L'URL du fournisseur d'identité OIDC qui permet au serveur d'API de découvrir les clés de signature publiques pour vérifier les jetons. L'URL doit commencer par `https://` et correspondre à la revendication `iss` dans les jetons d'ID OIDC du fournisseur. Conformément à la norme OIDC, les composants du chemin sont autorisés, mais pas les paramètres de requête. Généralement, l'URL se compose uniquement d'un nom d'hôte, comme `https://server.example.org` ou `https://example.com`. Cette URL doit pointer vers le niveau sous `.well-known/openid-configuration` et doit être accessible publiquement sur Internet.

ID client (également connu sous le nom depublic)

L'ID de l'application cliente qui envoie les demandes d'authentification au fournisseur d'identité OIDC.

Vous pouvez associer un fournisseur d'identité à l'aide de `eksctl` ou de la AWS Management Console.

eksctl

Pour associer un fournisseur d'identité OIDC à votre cluster avec **`eksctl`**

1. Créez un fichier nommé *`associate-identity-provider.yaml`* avec les contenus suivants. Remplacez les *`example values`* par vos propres valeurs. Les valeurs de la section `identityProviders` sont obtenues auprès de votre fournisseur d'identité OIDC. Des valeurs ne sont requises que pour les paramètres `name`, `type`, `issuerUrl` et `clientId` sous `identityProviders`.

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: your-region-code
```

```
identityProviders:
  - name: my-provider
    type: oidc
    issuerUrl: https://example.com
    clientId: kubernetes
    usernameClaim: email
    usernamePrefix: my-username-prefix
    groupsClaim: my-claim
    groupsPrefix: my-groups-prefix
    requiredClaims:
      string: string
    tags:
      env: dev
```

> **Important**
>
> Ne spécifiez pas de valeur `system:`, ou n'importe quelle partie de cette chaîne, pour `groupsPrefix` ou `usernamePrefix`.

2. Créez le fournisseur.

```
eksctl associate identityprovider -f associate-identity-provider.yaml
```

3. Pour utiliser `kubectl` afin de travailler avec votre cluster et votre fournisseur d'identité OIDC, consultez Utilisation de `kubectl` dans la documentation Kubernetes.

AWS Management Console

Pour associer un fournisseur OIDC d'identité à votre cluster à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Sélectionnez votre cluster, puis sélectionnez l'onglet Accès.
3. Dans la section Fournisseurs OIDC d'identité, sélectionnez Fournisseur d'identité associé.
4. Sur la page Associer le fournisseur d'identité OIDC, saisissez ou sélectionnez les options suivantes, puis sélectionnez Associer.

   - Dans Nom, saisissez un nom unique pour le fournisseur.

- Pour URL de l'émetteur, saisissez l'URL de votre fournisseur. Cette URL doit être accessible via Internet.
- Pour ID du client, saisissez l'ID de client du fournisseur d'identité OIDC (également appelé audience).
- Pour Nom d'utilisateur, saisissez la revendication à utiliser comme nom d'utilisateur.
- Pour Revendication de groupes, saisissez la revendication à utiliser comme groupe de l'utilisateur.
- (Facultatif) Sélectionnez Options avancées, puis saisissez ou sélectionnez les informations suivantes :
  - Nom d'utilisateur : saisissez un préfixe à ajouter aux revendications de nom d'utilisateur. Le préfixe est ajouté aux revendications de nom d'utilisateur pour éviter les conflits avec les noms existants. Si vous ne fournissez aucune valeur et que le nom d'utilisateur est une valeur autre que `email`, le préfixe correspond par défaut à la valeur URL de l'émetteur. Vous pouvez utiliser la valeur `-` pour désactiver toutes les préfixes. Ne spécifiez pas la valeur `system:` ou n'importe quelle partie de cette chaîne.
  - Préfixe de groupes : saisissez un préfixe à ajouter aux revendications de groupes. Le préfixe est ajouté aux revendications de groupe pour éviter les conflits avec des noms existants (tels que `system: groups`). Par exemple, la valeur `oidc:` crée des noms de groupe comme `oidc:engineering` et `oidc:infra`. Ne spécifiez pas la valeur `system:` ou n'importe quelle partie de cette chaîne.
  - Revendications requises : sélectionnez Ajouter une revendication et saisissez une ou plusieurs paires de valeurs clés qui décrivent les revendications requises dans le jeton d'ID de client. Les paires décrivent les revendications requises dans le jeton d'ID. Si elle est définie, la présence de chaque revendication est vérifiée dans le jeton d'ID avec une valeur correspondante.

5. Pour utiliser `kubectl` afin de travailler avec votre cluster et votre fournisseur d'identité OIDC, consultez Utilisation de `kubectl` dans la documentation Kubernetes.

Dissocier un fournisseur d'identité OIDC de votre cluster

Si vous dissociez un fournisseur d'identité OIDC de votre cluster, les utilisateurs inclus dans le fournisseur ne peuvent plus accéder au cluster. Cependant, vous pouvez toujours accéder au cluster avec les principaux IAM.

Pour dissocier un fournisseur d'identité OIDC de votre cluster à l'aide de la AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans la section Fournisseurs d'identité OIDC, sélectionnez Dissocier, saisissez le nom du fournisseur d'identité, puis sélectionnez `Disassociate`.

Exemple de politique IAM

Si vous voulez empêcher un fournisseur d'identité OIDC d'être associé à un cluster, créez et associez la politique IAM suivante aux comptes IAM de vos administrateurs Amazon EKS. Pour plus d'informations, consultez Création de politiques IAM et Ajout d'autorisations d'identité IAM dans le guide de l'utilisateur IAM et Actions, ressources et clés de condition pour Amazon Elastic Kubernetes Service dans la référence d'autorisation de service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "denyOIDC",
            "Effect": "Deny",
            "Action": [
                "eks:AssociateIdentityProviderConfig"
            ],
            "Resource": "arn:aws:eks:us-west-2.amazonaws.com:111122223333:cluster/*"

        },
        {
            "Sid": "eksAdmin",
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        }
    ]
}
```

L'exemple de politique suivant autorise l'association de fournisseur d'identité OIDC si l'`clientID` est `kubernetes` et l'`issuerUrl` est `https://cognito-idp.us-west-2amazonaws.com/*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCognitoOnly",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotLikeIfExists": {
                    "eks:issuerUrl": "https://cognito-idp.us-west-2.amazonaws.com/*"
                }
            }
        },
        {
            "Sid": "DenyOtherClients",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotEquals": {
                    "eks:clientId": "kubernetes"
                }
            }
        },
        {
            "Sid": "AllowOthers",
            "Effect": "Allow",
            "Action": "eks:*",
            "Resource": "*"
        }
    ]
}
```

Fournisseurs d'identité OIDC validés par des partenaires

Amazon EKS entretient des relations avec un réseau de partenaires qui offrent une assistance pour les fournisseurs d'identité OIDC compatibles. Consultez la documentation suivante des partenaires pour savoir comment intégrer le fournisseur d'identité à Amazon EKS.

Partenaire	Produit (langue française non garantie)	Documentation
PingIdentity	PingOne pour Enterprise	Instructions d'installation

Amazon EKS vise à vous offrir un large choix d'options pour couvrir tous les cas d'utilisation. Si vous développez un fournisseur d'identité OIDC compatible pris en charge sur le plan commercial qui n'est pas répertorié ici, contactez notre équipe partenaire à aws-container-partners@amazon.com pour plus d'informations.

Création ou mise à jour d'un fichier **kubeconfig** pour un cluster Amazon EKS

Dans cette rubrique, vous allez créer un fichier `kubeconfig` pour votre cluster (ou mettre à jour un fichier existant).

L'outil de ligne de commande `kubectl` utilise les informations de configuration dans les fichiers `kubeconfig` pour communiquer avec le serveur API d'un cluster. Pour plus d'informations, consultez Organizing Cluster Access Using kubeconfig Files (langue française non garantie) dans la documentation Kubernetes.

Amazon EKS utilise la commande `aws eks get-token` avec `kubectl` pour l'authentification du cluster. Par défaut, AWS CLI utilise les mêmes informations d'identification que celles renvoyées par la commande suivante :

```
aws sts get-caller-identity
```

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.

- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- Un utilisateur IAM ou un rôle IAM avec l'autorisation d'utiliser l'action d'API `eks:DescribeCluster` pour le cluster que vous spécifiez. Pour plus d'informations, consultez Exemples de politiques basées sur l'identité d'Amazon EKS. Si vous utilisez une identité de votre propre fournisseur OpenID Connect pour accéder à votre cluster, consultez la section Utilisation de `kubectl` dans la documentation Kubernetes pour créer ou mettre à jour votre fichier `kube config`.

Créer le fichier **`kubeconfig`** automatiquement

Prérequis

- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- Autorisation d'utiliser l'action d'API `eks:DescribeCluster` pour le cluster que vous spécifiez. Pour plus d'informations, consultez Exemples de politiques basées sur l'identité d'Amazon EKS.

Pour créer votre **kubeconfig** fichier à l'aide du AWS CLI

1. Créez ou mettez à jour un fichier kubeconfig pour votre cluster. *Remplacez le code de région par le nom de votre Région AWS cluster.*

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Par défaut, le fichier de configuration résultant est créé dans le chemin kubeconfig par défaut (.kube) dans votre répertoire de base ou fusionné avec un fichier config existant à cet emplacement. Vous pouvez spécifier un autre chemin avec l'option **--kubeconfig**.

   Vous pouvez spécifier un ARN de rôle IAM avec l'option **--role-arn** à utiliser pour l'authentification lorsque vous émettez des commandes kubectl. Dans le cas contraire, le principal IAM de votre chaîne d'identification par défaut AWS CLI ou du SDK est utilisé. Vous pouvez consulter votre identité par défaut AWS CLI ou celle de votre SDK en exécutant la aws sts get-caller-identity commande.

   Pour connaître toutes les options disponibles, exécutez la commande aws eks update-kubeconfig help ou consultez update-kubeconfig dans la Référence des commandes de l'AWS CLI .

2. Testez votre configuration.

   ```
   kubectl get svc
   ```

   L'exemple qui suit illustre un résultat.

   ```
   NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
   svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
   ```

   Si vous recevez d'autres erreurs concernant les types d'autorisations ou de ressources, consultez Accès non autorisé ou refusé (kubectl) dans la rubrique relative à la résolution des problèmes.

Autoriser les charges de travail Kubernetes à utiliser les comptes de service AWSKubernetes

Un compte de service Kubernetes fournit une identité pour les processus exécutés dans un Pod. Pour plus d'informations, consultez la rubrique Gestion des comptes de service dans la documentation Kubernetes. Si vous avez Pod besoin d'accéder à AWS des services, vous pouvez associer le compte de service à une AWS Identity and Access Management identité pour accorder cet accès. Pour plus d'informations, consultez Rôles IAM pour les comptes de service.

Jetons de compte de service

La fonctionnalité `BoundServiceAccountTokenVolume` est activée par défaut dans les versions Kubernetes. Cette fonctionnalité améliore la sécurité des jetons de compte de service en permettant aux charges de travail s'exécutant sur Kubernetes de demander des jetons web JSON liés au public ciblé, à l'heure et aux clés. Les jetons de compte de service ont une expiration d'une heure. Dans les versions antérieures de Kubernetes, les jetons n'avaient pas de période d'expiration. Cela signifie que les clients qui exploitent ces jetons doivent actualiser les jetons en moins d'une heure. Les kits SDK clients Kubernetes suivants actualisent automatiquement les jetons dans le délai requis :

- Go version `0.15.7` et ultérieure
- Python version `12.0.0` et ultérieure
- Java version `9.0.0` et ultérieure
- JavaScript version `0.10.3` et versions ultérieures
- Branche `master` Ruby
- Haskell version `0.3.0.0`
- C# version `7.0.5` et ultérieure

Si votre charge de travail utilise une version client antérieure, vous devez la mettre à jour. Pour garantir une migration en douceur des clients vers les nouveaux jetons de compte de service limités dans le temps, Kubernetes ajoute une période d'expiration prolongée au jeton de compte de service, en plus de la période par défaut d'une heure. Pour les clusters Amazon EKS, la période d'expiration prolongée est de 90 jours. Le serveur d'API Kubernetes de votre cluster Amazon EKS rejette les demandes dont les jetons datent de plus de 90 jours. Nous vous recommandons de vérifier vos applications et leurs dépendances pour vous assurer que les SDK des clients Kubernetes sont de versions identiques ou ultérieures aux versions répertoriées ci-dessus.

Lorsque le serveur API reçoit des requêtes avec des jetons de plus d'une heure, il annote l'événement du journal d'audit de l'API avec `annotations.authentication.k8s.io/stale-token`. La valeur de l'annotation ressemble à l'exemple suivant :

```
subject: system:serviceaccount:common:fluent-bit, seconds after warning threshold:
 4185802.
```

Si votre cluster a une journalisation de plan de contrôle activée, les annotations se trouvent dans les journaux d'audit. Vous pouvez utiliser la requête CloudWatch Logs Insights suivante pour identifier tous les éléments Pods de votre cluster Amazon EKS qui utilisent des jetons périmés :

```
fields @timestamp
| filter @logStream like /kube-apiserver-audit/
| filter @message like /seconds after warning threshold/
| parse @message "subject: *, seconds after warning threshold:*\"" as subject,
 elapsedtime
```

Le `subject` fait référence au compte de service utilisé par le Pod. Le `elapsedtime` indique le temps écoulé (en secondes) après la lecture du dernier jeton. Les demandes adressées au serveur d'API sont refusées lorsque le `elapsedtime` dépasse 90 jours (7 776 000 secondes). Vous devez mettre à jour de manière proactive le kit SDK client Kubernetes de vos applications pour utiliser l'une des versions répertoriées ci-dessus qui actualise automatiquement le jeton. Si le jeton de compte de service utilisé dure près de 90 jours et que vous n'avez pas suffisamment de temps pour mettre à jour les versions de votre SDK client avant l'expiration du jeton, vous pouvez résilier les Pods existants et en créer de nouveaux. Il en résulte une récupération du jeton de compte de service, ce qui vous donne 90 jours supplémentaires pour mettre à jour les kits SDK de votre version client.

Si le Pod fait partie d'un déploiement, pour résilier les Pods tout en conservant une haute disponibilité, il est suggéré d'effectuer un déploiement à l'aide de la commande suivante. Remplacez *`my-deployment`* par le nom de votre déploiement.

```
kubectl rollout restart deployment/my-deployment
```

Modules complémentaires de cluster

Les modules complémentaires de cluster suivants ont été mis à jour pour utiliser le SDK client Kubernetes qui récupère automatiquement les jetons de compte de service. Nous vous

recommandons de vous assurer que les versions répertoriées, ou des versions ultérieures, sont installées sur votre cluster.

- Version de Amazon VPC CNI plugin for Kubernetes et des plug-ins d'aide aux métriques `1.8.0` et version ultérieure. Pour vérifier votre version actuelle ou la mettre à jour, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS et cni-metrics-helper.
- CoreDNS version `1.8.4` et ultérieure Pour vérifier votre version actuelle ou la mettre à jour, consultez Utilisation du module complémentaire CoreDNS Amazon EKS.
- AWS Load Balancer Controller version `2.0.0` et ultérieure Pour vérifier votre version actuelle ou la mettre à jour, consultez Qu'est-ce que AWS Load Balancer Controller ?.
- Une version actuelle de `kube-proxy`. Pour vérifier votre version actuelle ou la mettre à jour, consultez Utilisation du module complémentaire Kubernetes `kube-proxy`.
- AWS pour la version Fluent Bit `2.25.0` ou ultérieure. Pour mettre à jour votre version actuelle, consultez Versions sur GitHub.
- Version d'image Fluentd 1.14.6-1.2 ou version ultérieure et plug-in de filtre Fluentd pour les métadonnées de Kubernetes version 2.11.1 ou ultérieure.

Octroi AWS Identity and Access Management d'autorisations aux charges de travail sur les clusters Amazon Elastic Kubernetes Service

Amazon EKS propose deux méthodes pour accorder AWS Identity and Access Management des autorisations aux charges de travail exécutées dans des clusters Amazon EKS : les rôles IAM pour les comptes de service et les identités EKS Pod.

Rôles IAM pour les comptes de service

> Les rôles IAM pour les comptes de service (IRSA) configurent les applications Kubernetes exécutées AWS avec des autorisations IAM précises pour accéder à diverses autres ressources telles que les compartiments Amazon AWS S3, les tables Amazon DynamoDB, etc. Vous pouvez exécuter plusieurs applications ensemble dans le même cluster Amazon EKS, et vous assurer que chaque application ne dispose que de l'ensemble minimum d'autorisations dont elle a besoin. L'IRSA a été conçu pour prendre en charge diverses options de Kubernetes déploiement prises en charge par AWS Amazon EKS, Amazon EKS Anywhere et les Kubernetes clusters autogérés sur les instances Amazon EC2. Red Hat OpenShift Service on AWS Ainsi, l'IRSA a été créée à

l'aide d'un AWS service de base tel que IAM, et ne dépendait pas directement du service Amazon EKS et de l'API EKS. Pour plus d'informations, consultez Rôles IAM pour les comptes de service.

Identités du pod EKS

EKS Pod Identity offre aux administrateurs de clusters un flux de travail simplifié pour authentifier les applications afin d'accéder à diverses autres AWS ressources telles que les compartiments Amazon S3, les tables Amazon DynamoDB, etc. L'identité du pod EKS est réservée uniquement à EKS et, par conséquent, elle simplifie la manière dont les administrateurs de cluster peuvent configurer les applications Kubernetes pour obtenir des autorisations IAM. Ces autorisations peuvent désormais être facilement configurées en moins d'étapes directement via l'API EKS AWS Management Console AWS CLI, et il n'y a aucune action à effectuer au sein du cluster dans aucun Kubernetes objet. Les administrateurs de cluster n'ont pas besoin de basculer entre les services EKS et IAM, ni d'utiliser des opérations IAM privilégiées pour configurer les autorisations requises par vos applications. Les rôles IAM peuvent désormais être utilisés dans plusieurs clusters sans qu'il soit nécessaire de mettre à jour la politique d'approbation des rôles lors de la création de nouveaux clusters. Les informations d'identification IAM fournies par l'identité du pod EKS comprennent des balises de session de rôle, avec des attributs tels que le nom du cluster, l'espace de noms, le nom du compte de service. Les balises de session de rôle permettent aux administrateurs de créer un rôle unique qui peut fonctionner avec tous les comptes de service en autorisant l'accès aux AWS ressources en fonction des balises correspondantes. Pour plus d'informations, consultez Identités du pod EKS.

Comparaison de l'identité du pod EKS et de l'IRSA

À un niveau élevé, l'identité du pod EKS et l'IRSA vous permettent tous deux d'accorder des autorisations IAM aux applications exécutées sur des clusters Kubernetes. Mais ils sont fondamentalement différents dans la façon dont vous les configurez, les limites prises en charge et les fonctionnalités activées. Ci-dessous, nous comparons certaines des facettes clés des deux solutions.

	Identité du pod EKS	IRSA
Extensibilité des rôles	Vous devez configurer chaque rôle une fois pour établir la confiance avec le nouveau principal de service Amazon EKS `pods.eks.amazonaws`	Vous devez mettre à jour la politique d'approbation du rôle IAM avec le nouveau point de terminaison du fournisseur de cluster EKS OIDC chaque fois

	Identité du pod EKS	IRSA
	`.com` . Après cette étape unique, vous n'avez pas besoin de mettre à jour la politique d'approbation du rôle chaque fois qu'il est utilisé dans un nouveau cluster.	que vous voulez utiliser le rôle dans un nouveau cluster.
Capacité de mise à l'échelle des clusters	L'identité du pod EKS n'exige pas que les utilisateurs configurent le fournisseur IAM OIDC, de sorte que cette limite ne s'applique pas.	Chaque cluster EKS possède une URL de diffuseur OpenID Connect (OIDC) qui lui est associée. Pour utiliser IRSA, un fournisseur OpenID Connect unique doit être créé pour chaque cluster EKS dans IAM. IAM a une limite globale par défaut de 100 fournisseurs OIDC pour chaque Compte AWS. Si vous prévoyez d'avoir plus de 100 clusters EKS pour chacun Compte AWS avec IRSA, vous atteindrez la limite de OIDC fournisseurs IAM.

	Identité du pod EKS	IRSA
Capacité de mise à l'échelle des rôles	L'identité du pod EKS n'exige pas que les utilisateurs définissent la relation d'approbation entre le rôle IAM et le compte de service dans la politique d'approbation, de sorte que cette limite ne s'applique pas.	Dans IRSA, vous définissez la relation d'approbation entre un rôle IAM et un compte de service dans la politique d'approbation du rôle. Par défaut, la longueur de la politique d'approbation est `2048`. Cela signifie que vous pouvez généralement définir 4 relations d'approbation dans une seule politique d'approbation. Bien qu'il soit possible d'augmenter la limite de longueur de la politique d'approbation, vous êtes généralement limité à un maximum de 8 relations d'approbation au sein d'une même politique d'approbation.

	Identité du pod EKS	IRSA
Réutilisation des rôles	AWS STS les informations d'identification temporair es fournies par EKS Pod Identity incluent les balises de session de rôle, telles que le nom du cluster, l'espace de noms, le nom du compte de service. Les balises de session de rôle permettent aux administrateurs de créer un rôle IAM unique qui peut être utilisé avec plusieurs comptes de service, avec différent es autorisations effective s, en autorisant l'accès aux ressources AWS en fonction des balises qui leur sont attachées. C'est ce qu'on appelle le contrôle d'accès par attributs (ABAC). Pour plus d'informations, consultez Définition des autorisations pour que les identités du pod EKS assument des rôles en fonction des balises.	AWS STS les balises de session ne sont pas prises en charge. Vous pouvez réutiliser un rôle entre les clusters, mais chaque pod reçoit toutes les autorisations du rôle.
Environnements pris en charge	L'identité du pod EKS est uniquement disponible sur Amazon EKS.	L'IRSA peut être utilisé comme Amazon EKS, Amazon EKS Anywhere et des Kubernete s clusters autogérés sur des instances Amazon EC2. Red Hat OpenShift Service on AWS

	Identité du pod EKS	IRSA
Versions EKS prises en charge	EKS Kubernetes versions `1.24` ou ultérieures. Pour les versions de plateform e spécifiques, consultez Versions du cluster de l'identit é du pod EKS.	Toutes les versions de clusters EKS prises en charge.

Identités du pod EKS

Les applications situées dans les conteneurs Pod d'un peuvent utiliser un AWS SDK ou le AWS CLI pour envoyer des demandes d'API aux autorisations Services AWS Using AWS Identity and Access Management (IAM). Les applications doivent signer leurs demandes AWS d'API avec des AWS informations d'identification.

Les identités du pod EKS permettent de gérer les informations d'identification de vos applications, de la même manière que les profils d'instance Amazon EC2 fournissent des informations d'identification aux instances Amazon EC2. Au lieu de créer et de distribuer vos AWS informations d'identification aux conteneurs ou d'utiliser le rôle de l'instance Amazon EC2, vous associez un rôle IAM à un compte de Kubernetes service et vous le configurez Pods pour utiliser le compte de service.

Chaque identité du pod EKS mappe un rôle à un compte de service dans un espace de noms dans le cluster spécifié. Si vous avez la même application dans plusieurs clusters, vous pouvez faire des associations identiques dans chaque cluster sans modifier la politique d'approbation du rôle.

Si un pod utilise un compte de service qui a une association, Amazon EKS définit des variables d'environnement dans les conteneurs du pod. Les variables d'environnement configurent les AWS SDK, y compris le AWS CLI, pour utiliser les informations d'identification EKS Pod Identity.

Avantages des identités du pod EKS

Les identités du pod EKS offrent les avantages suivants :

- Moindre privilège – Vous pouvez définir les autorisations IAM sur un compte de service et seuls les Pods qui utilisent ce compte de service ont accès à ces autorisations. Cette fonctionnalité élimine également le besoin de solutions tierces telles que `kiam` ou `kube2iam`.

- Isolement des informations d'identification : les conteneurs d'un Pod's peuvent uniquement récupérer les informations d'identification du rôle IAM associées au compte de service que le conteneur utilise. Un conteneur n'a jamais accès aux informations d'identification qui sont utilisées par d'autres conteneurs dans d'autres Pods. Lors de l'utilisation des identités du pod, les conteneurs Pod's disposent également des autorisations attribuées au rôle IAM de nœud Amazon EKS, sauf si vous bloquez l'accès du Pod au service de métadonnées d'instance (IMDS) Amazon EC2. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.
- Auditabilité — La journalisation des accès et des événements est disponible AWS CloudTrail pour faciliter l'audit rétrospectif.

L'identité du pod EKS est une méthode plus simple que Rôles IAM pour les comptes de service, car cette méthode n'utilise pas de fournisseurs d'identité OIDC. L'identité du pod EKS présente les améliorations suivantes :

- Opérations indépendantes : dans de nombreuses organisations, la création de fournisseurs d'identité OIDC relève d'équipes différentes de l'administration des clusters Kubernetes. L'identité du pod EKS présente une séparation nette des tâches, où toute la configuration des associations de l'identité du pod EKS est effectuée dans Amazon EKS et toute la configuration des autorisations IAM est effectuée dans IAM.
- Réutilisabilité : l'identité du pod EKS utilise un seul principal IAM au lieu des principaux distincts pour chaque cluster que les rôles IAM des comptes de service utilisent. Votre administrateur IAM ajoute le principal suivant à la politique d'approbation de n'importe quel rôle pour le rendre utilisable par les identités du pod EKS.

```
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            }
```

- Évolutivité — Chaque ensemble d'informations d'identification temporaires est utilisé par le EKS Auth service dans EKS Pod Identity, et non par chaque AWS SDK que vous exécutez dans chaque module. Ensuite, l'agent de l'identité du pod Amazon EKS qui s'exécute sur chaque nœud émet les informations d'identification aux kits SDK. Ainsi, la charge est réduite à une fois pour chaque nœud et n'est pas dupliquée dans chaque pod. Pour plus de détails sur le processus, consultez Fonctionnement de l'identité du pod EKS.

Pour plus d'informations sur la comparaison des deux solutions, consultez Autoriser les charges de travail Kubernetes à utiliser les comptes de service AWSKubernetes.

Présentation de la configuration des identités du pod EKS

Activez les identités du pod EKS en effectuant les procédures suivantes :

1. Configuration de l'agent d'identité Amazon EKS Pod — Vous n'effectuez cette procédure qu'une seule fois pour chaque cluster.
2. Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM avec EKS Pod Identity — Suivez cette procédure pour chaque ensemble unique d'autorisations que vous souhaitez attribuer à une application.
3. Configurer Pods pour utiliser un compte Kubernetes de service— Complétez cette procédure pour tous ceux Pod qui ont besoin d'accéder à Services AWS.
4. Utiliser un AWS SDK compatible— Vérifiez que la charge de travail utilise un AWS SDK d'une version prise en charge et qu'elle utilise la chaîne d'identification par défaut.

Considérations relatives à l'identité du pod EKS

- Vous pouvez associer un rôle IAM à chaque compte de service Kubernetes dans chaque cluster. Vous pouvez modifier le rôle mappé au compte de service en modifiant l'association des identités du pod EKS.
- Vous ne pouvez associer que des rôles identiques à Compte AWS ceux du cluster. Vous pouvez déléguer l'accès d'un autre compte au rôle de ce compte que vous configurez pour que les identités du pod EKS utilisent. Pour un didacticiel sur la délégation d'accès`AssumeRole`, voir Déléguer l'accès entre AWS comptes à l'aide de rôles IAM dans le Guide de l'utilisateur IAM.
- L'agent d'identité du pod EKS est nécessaire. Il s'exécute en tant que Kubernetes `DaemonSet` sur vos nœuds et ne fournit des informations d'identification qu'aux pods du nœud sur lequel il s'exécute. Pour plus d'informations sur la compatibilité de l'agent d'identité du pod EKS, consultez la section suivante Restrictions relatives à l'identité du pod EKS.
- L'agent d'identité du pod EKS utilise le `hostNetwork` du nœud et il utilise le port `80` et le port `2703` sur une adresse locale de liaison sur le nœud. Cette adresse est `169.254.170.23` pour les clusters IPv4 et `[fd00:ec2::23]` pour les clusters IPv6.

  Si vous désactivez `IPv6` des adresses ou si vous empêchez les adresses `IPv6` IP de l'hôte local, l'agent ne peut pas démarrer. Pour démarrer l'agent sur des nœuds qui ne peuvent pas être

utilisésIPv6, suivez les étapes décrites Désactiver IPv6 dans l'agent d'identité EKS Pod pour désactiver la IPv6 configuration.

Versions du cluster de l'identité du pod EKS

Pour utiliser les identités du pod EKS, le cluster doit avoir une version de plateforme identique ou ultérieure à la version répertoriée dans le tableau suivant, ou une version Kubernetes postérieure aux versions répertoriées dans le tableau.

Version de Kubernetes	Version de plateforme
1.30	eks.2
1.29	eks.1
1.28	eks.4
1.27	eks.8
1.26	eks.9
1.25	eks.10
1.24	eks.13

Versions complémentaires compatibles avec EKS Pod Identity

> **Important**
>
> Pour utiliser EKS Pod Identity avec un module complémentaire EKS, vous devez créer l'association EKS Pod Identity manuellement. Ne choisissez pas de rôle IAM dans la configuration du module complémentaire dans le AWS Management Console, ce rôle n'est utilisé qu'avec IRSA.

Les modules complémentaires Amazon EKS et les modules complémentaires autogérés qui nécessitent des informations d'identification IAM peuvent utiliser EKS Pod Identity, IRSA ou le rôle

d'instance. La liste des modules complémentaires qui utilisent les informations d'identification IAM compatibles avec EKS Pod Identity est la suivante :

- Amazon VPC CNI plugin for Kubernetes`1.15.5-eksbuild.1`ou plus tard
- AWS Load Balancer Controller`2.7.0`ou plus tard. Notez que le AWS Load Balancer Controller n'est pas disponible en tant que module complémentaire EKS, mais qu'il est disponible en tant que module complémentaire autogéré.

Restrictions relatives à l'identité du pod EKS

Les identités du pod EKS sont disponibles sur les éléments suivants :

- Les versions du cluster Amazon EKS répertoriées dans la rubrique précédente Versions du cluster de l'identité du pod EKS.
- Les composants master du cluster qui sont des instances Amazon EC2 Linux.

Les identités du pod EKS ne sont pas disponibles sur ce qui suit :

- Régions chinoises.
- AWS GovCloud (US).
- AWS Outposts.
- Amazon EKS Anywhere.
- Clusters Kubernetes que vous créez et exécutez sur Amazon EC2. Les composants de l'identité du pod Amazon EKS sont uniquement disponibles sur Amazon EKS.

Vous ne pouvez pas utiliser les identités du pod EKS avec :

- Les pods qui s'exécutent n'importe où, à l'exception des instances Amazon EC2 Linux. Les pods Linux et Windows qui s'exécutent sur AWS Fargate (Fargate) ne sont pas pris en charge. Les pods qui s'exécutent sur des instances Amazon EC2 Windows ne sont pas pris en charge.
- Modules complémentaires Amazon EKS nécessitant des informations d'identification IAM. Les modules complémentaires EKS ne peuvent utiliser que des rôles IAM des comptes de service à la place. La liste des modules complémentaires EKS qui utilisent des informations d'identification IAM comprend :
  - Les pilotes CSI de stockage : EBS CSI, EFS CSI, pilote Amazon FSx for Lustre CSI, pilote Amazon FSx pour NetApp ONTAP CSI, pilote Amazon FSx pour OpenZFS CSI, pilote Amazon

File Cache CSI, fournisseur de secrets et de configuration (ASCP) pour le pilote AWS CSI Secrets Store Kubernetes

> **Note**
>
> Si ces contrôleurs, pilotes et plug-ins sont installés en tant que modules complémentaires autogérés au lieu de modules complémentaires EKS, ils prennent en charge EKS Pod Identities à condition qu'ils soient mis à jour pour utiliser les derniers AWS SDK.

Fonctionnement de l'identité du pod EKS

Les associations de l'identité du pod Amazon EKS offrent une capacité de gestion des informations d'identification à utiliser pour les applications, de la même façon que les profils d'instance Amazon EC2 fournissent des informations d'identification aux instances EC2.

L'identité du pod Amazon EKS fournit des informations d'identification à vos charges de travail avec une API d'authentification EKS supplémentaire et un pod d'agent qui s'exécute sur chaque nœud.

Dans vos modules complémentaires, tels que les modules complémentaires Amazon EKS, le contrôleur autogéré, les opérateurs et les autres modules complémentaires, l'auteur doit mettre à jour son logiciel pour utiliser les derniers AWS SDK. Pour la liste de compatibilité entre l'identité du pod EKS et les modules complémentaires produits par Amazon EKS, reportez-vous à la section précédente Restrictions relatives à l'identité du pod EKS.

Utilisation des identités du pod EKS dans votre code

Dans votre code, vous pouvez utiliser les AWS SDK pour accéder aux AWS services. Vous écrivez du code pour créer un client pour un AWS service avec un SDK, et par défaut, le SDK recherche les informations d' AWS Identity and Access Management identification à utiliser dans une chaîne d'emplacements. Une fois les informations d'identification valides trouvées, la recherche s'arrête. Pour plus d'informations sur les emplacements par défaut utilisés, consultez la chaîne de fournisseurs d'informations d'identification dans le Guide de référence AWS des SDK et des outils.

Les identités du pod EKS ont été ajoutées au fournisseur d'informations d'identification du conteneur qui est recherché dans une étape de la chaîne d'informations d'identification par défaut. Si vos charges de travail utilisent actuellement des informations d'identification antérieures dans la chaîne de fournisseurs d'informations d'identification, ces informations continueront d'être utilisées même si vous configurez une association d'identité du pod EKS pour la même charge de travail. Ainsi,

vous pouvez migrer en toute sécurité à partir d'autres types d'informations d'identification en créant d'abord l'association, avant de supprimer les anciennes informations d'identification.

Le fournisseur d'informations d'identification de conteneur fournit des informations d'identification temporaires à partir d'un agent qui s'exécute sur chaque nœud. Dans Amazon EKS, l'agent est l'agent d'identité du pod Amazon EKS et sur Amazon Elastic Container Service, l'agent est le `amazon-ecs-agent`. Les kits SDK utilisent des variables d'environnement pour localiser l'agent auquel se connecter.

En revanche, les rôles IAM pour les comptes de service fournissent un jeton d'identité Web que le AWS SDK doit échanger avec AWS Security Token Service . `AssumeRoleWithWebIdentity`

Fonctionnement de l'agent d'identité du pod EKS avec un Pod

1. Lorsque Amazon EKS démarre un nouveau pod qui utilise un compte de service avec une association Identité du pod EKS, le cluster ajoute le contenu suivant au manifeste Pod :

```
    env:
    - name: AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE
      value: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/eks-pod-
identity-token"
    - name: AWS_CONTAINER_CREDENTIALS_FULL_URI
      value: "http://169.254.170.23/v1/credentials"
    volumeMounts:
    - mountPath: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/"
      name: eks-pod-identity-token
  volumes:
  - name: eks-pod-identity-token
    projected:
      defaultMode: 420
      sources:
      - serviceAccountToken:
          audience: pods.eks.amazonaws.com
          expirationSeconds: 86400 # 24 hours
          path: eks-pod-identity-token
```

2. Kubernetes sélectionne le nœud sur lequel exécuter le pod. Ensuite, l'agent Amazon EKS Pod Identity sur le nœud utilise l'AssumeRoleForPodIdentityaction pour récupérer des informations d'identification temporaires à partir de l'API EKS Auth.
3. L'agent EKS Pod Identity met ces informations d'identification à la disposition AWS des SDK que vous exécutez dans vos conteneurs.

4. Vous utilisez le kit SDK dans votre application sans spécifier de fournisseur d'informations d'identification pour utiliser la chaîne d'informations d'identification par défaut. Vous pouvez également spécifier le fournisseur d'informations d'identification du conteneur. Pour plus d'informations sur les emplacements par défaut utilisés, consultez la chaîne de fournisseurs d'informations d'identification dans le Guide de référence AWS des SDK et des outils.
5. Le kit SDK utilise les variables d'environnement pour se connecter à l'agent d'identité du pod EKS et récupérer les informations d'identification.

> **Note**
>
> Si vos charges de travail utilisent actuellement des informations d'identification antérieures dans la chaîne de fournisseurs d'informations d'identification, ces informations continueront d'être utilisées même si vous configurez une association d'identité du pod EKS pour la même charge de travail.

Configuration de l'agent d'identité Amazon EKS Pod

Les associations de l'identité du pod Amazon EKS offrent une capacité de gestion des informations d'identification à utiliser pour les applications, de la même façon que les profils d'instance Amazon EC2 fournissent des informations d'identification aux instances EC2.

L'identité du pod Amazon EKS fournit des informations d'identification à vos charges de travail avec une API d'authentification EKS supplémentaire et un pod d'agent qui s'exécute sur chaque nœud.

Considérations

- **IPv6**

  Par défaut, l'agent EKS Pod Identity écoute sur une `IPv6` adresse `IPv4` et les pods pour demander des informations d'identification. L'agent utilise l'adresse IP de bouclage (localhost) `169.254.170.23` pour `IPv4` et l'adresse IP localhost pour. `[fd00:ec2::23]` `IPv6`

  Si vous désactivez `IPv6` des adresses ou si vous empêchez les adresses `IPv6` IP de l'hôte local, l'agent ne peut pas démarrer. Pour démarrer l'agent sur des nœuds qui ne peuvent pas être utilisés`IPv6`, suivez les étapes décrites Désactiver `IPv6` dans l'agent d'identité EKS Pod pour désactiver la `IPv6` configuration.

Création d'un agent d'identité du pod Amazon EKS

Conditions préalables pour l'agent

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS. La version du cluster et la version de plateforme doivent être identiques ou ultérieures aux versions répertoriées dans Versions du cluster de l'identité du pod EKS.
- Le rôle de nœud dispose d'autorisations permettant à l'agent d'effectuer l'action `AssumeRoleForPodIdentity` dans l'API d'authentification EKS. Vous pouvez utiliser l'action AWS politique gérée : Amazoneks WorkerNodePolicy ou ajouter une politique personnalisée similaire à la suivante :

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

  Cette action peut être limitée par des balises pour restreindre les rôles qui peuvent être assumés par les pods qui utilisent l'agent.
- Les nœuds peuvent atteindre et télécharger des images à partir d'Amazon ECR. L'image du conteneur pour le module complémentaire se trouve dans les registres répertoriés dans Registres d'images de conteneur Amazon.

  Notez que vous pouvez modifier l'emplacement de l'image et `imagePullSecrets` prévoir des modules complémentaires EKS dans les paramètres de configuration facultatifs du AWS Management Console, et `--configuration-values` dans le AWS CLI.
- Les nœuds peuvent accéder à l'API d'authentification Amazon EKS. Pour les clusters privés, le `eks-auth` point de terminaison AWS PrivateLink est obligatoire.

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire l'agent d'identité du pod EKS.
3. Choisissez l'onglet Modules complémentaires.
4. Choisissez Obtenez plus de modules complémentaires.
5. Cochez la case en haut à droite du module complémentaire de l'agent d'identité du pod EKS, puis sélectionnez Suivant.
6. Sur la page Configurer les paramètres de modules complémentaires sélectionnés, sélectionnez n'importe quelle Version dans la liste déroulante Version.
7. (Facultatif) Développez les Paramètres de configuration facultatifs pour entrer une configuration supplémentaire. Par exemple, vous pouvez fournir un autre emplacement d'image de conteneur et `ImagePullSecrets`. Le JSON Schema avec les clés acceptées est indiqué dans Schéma de configuration du module complémentaire.

   Saisissez les clés et les valeurs de configuration dans Valeurs de configuration.
8. Choisissez Suivant.
9. Confirmez que les pods de l'agent d'identité du pod EKS sont en cours d'exécution sur votre cluster.

   ```
   kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
   ```

   L'exemple qui suit illustre un résultat.

   ```
   eks-pod-identity-agent-gmqp7                                          1/1
    Running   1 (24h ago)   24h
   eks-pod-identity-agent-prnsh                                          1/1
    Running   1 (24h ago)   24h
   ```

   Vous pouvez maintenant utiliser les associations d'identité du pod EKS dans votre cluster. Pour plus d'informations, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM avec EKS Pod Identity.

AWS CLI

1. Exécutez la AWS CLI commande suivante. Remplacez `my-cluster` par le nom de votre cluster.

```
aws eks create-addon --cluster-name my-cluster --addon-name eks-pod-identity-
agent --addon-version v1.0.0-eksbuild.1
```

> **Note**
>
> L'agent d'identité du pod EKS n'utilise pas le `service-account-role-arn` pour les rôles IAM des comptes de service. Vous devez fournir à l'agent d'identité du pod EKS des autorisations dans le rôle de nœud.

2. Confirmez que les pods de l'agent d'identité du pod EKS sont en cours d'exécution sur votre cluster.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

L'exemple qui suit illustre un résultat.

```
eks-pod-identity-agent-gmqp7                                          1/1
 Running   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                          1/1
 Running   1 (24h ago)   24h
```

Vous pouvez maintenant utiliser les associations d'identité du pod EKS dans votre cluster. Pour plus d'informations, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM avec EKS Pod Identity.

Mise à jour de l'agent d'identité du pod Amazon EKS

Mettez à jour le type de module complémentaire Amazon EKS. Si vous n'avez pas ajouté le type Amazon EKS du module complémentaire à votre cluster, consultez Création d'un agent d'identité du pod Amazon EKS.

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire l'agent d'identité du pod EKS.
3. Choisissez l'onglet Modules complémentaires.
4. Si une nouvelle version du module complémentaire est disponible, l'agent d'identité du pod EKS dispose d'un bouton Mettre à jour la version. Sélectionnez Mettre à jour la version.
5. Sur la page Configurer l'agent d'identité du pod Amazon EKS, sélectionnez la nouvelle version dans la liste déroulante Version.
6. Sélectionnez Enregistrer les modifications.

   La mise à jour peut prendre plusieurs secondes. Ensuite, vérifiez que la version du module complémentaire a été mise à jour en contrôlant le Statut.

AWS CLI

1. Déterminez la version du module complémentaire actuellement installée sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --query "addon.addonVersion" --output text
   ```

   L'exemple qui suit illustre un résultat.

   ```
   v1.0.0-eksbuild.1
   ```

   Vous devez créer le module complémentaire avant de pouvoir le mettre à jour à l'aide de cette procédure.
2. Mettez à jour votre module complémentaire à l'aide de la AWS CLI. Si vous souhaitez utiliser la AWS Management Console ou `eksctl` pour mettre à jour le module complémentaire, consultez la rubrique Mise à jour d'un module complémentaire. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée.

- Remplacez *my-cluster* par le nom de votre cluster.
- Remplacez *v1.0.0-eksbuild.1* par la version que vous souhaitez.
- Remplacez *111122223333* par votre ID de compte.
- Exécutez la commande suivante :

```
aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --addon-version v1.0.0-eksbuild.1'
```

La mise à jour peut prendre plusieurs secondes.

3. Assurez-vous que la version du module complémentaire a été mise à jour. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent
```

La mise à jour peut prendre plusieurs secondes.

L'exemple qui suit illustre un résultat.

```
{
    "addon": {
        "addonName": "eks-pod-identity-agent",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.0.0-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/eks-pod-identity-agent/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "tags": {}
    }
}
```

Configuration de l'agent d'identité EKS Pod

Désactiver **IPv6** dans l'agent d'identité EKS Pod

AWS Management Console

Désactiver **IPv6** dans AWS Management Console

1. Pour le désactiver `IPv6` dans l'agent EKS Pod Identity, ajoutez la configuration suivante aux paramètres de configuration facultatifs du module complémentaire EKS.

   a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

   b. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire.

   c. Choisissez l'onglet Modules complémentaires.

   d. Cochez la case en haut à droite du module complémentaire EKS Pod Identity Agent, puis choisissez Modifier.

   e. Sur la page Configurer l'agent d'identité EKS Pod :

      i. Sélectionnez la version que vous souhaitez utiliser. Nous vous recommandons de conserver la même version que celle de l'étape précédente et de mettre à jour la version et la configuration dans le cadre d'actions distinctes.

      ii. Sélectionnez Paramètres de configuration facultatifs.

      iii. Entrez la clé JSON `"agent":` et la valeur d'un objet JSON imbriqué avec une clé `"additionalArgs":` dans Valeurs de configuration. Le texte obtenu doit être un objet JSON valide. Si cette clé et cette valeur sont les seules données de la zone de texte, entourez-les d'accolades `{}`. L'exemple suivant montre que la politique réseau est activée :

```
{
    "agent": {
        "additionalArgs": {
            "-b": "169.254.170.23"
        }
    }
}
```

Cette configuration définit l'`IPv4`adresse comme étant la seule adresse utilisée par l'agent.

f. Pour appliquer la nouvelle configuration en remplaçant les modules EKS Pod Identity Agent, choisissez Enregistrer les modifications.

Amazon EKS applique les modifications aux modules complémentaires EKS en déployant l'agent d'identité Kubernetes `DaemonSet` pour EKS Pod. Vous pouvez suivre l'état du déploiement dans l'historique des mises à jour du module complémentaire dans AWS Management Console et avec`kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`.

`kubectl rollout`possède les commandes suivantes :

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Si le déploiement prend trop de temps, Amazon EKS annulera le déploiement et un message indiquant le type de mise à jour de l'extension et le statut Echec sera ajouté à l'historique des mises à jour de l'extension. Pour étudier les problèmes éventuels, commencez par l'historique du déploiement et exécutez-le `kubectl logs` sur un pod EKS Pod Identity Agent pour consulter les journaux d'EKS Pod Identity Agent.

2. Si la nouvelle entrée dans l'historique des mises à jour a le statut Successful, le déploiement est terminé et le module complémentaire utilise la nouvelle configuration dans tous les pods EKS Pod Identity Agent.

AWS CLI

Désactiver **`IPv6`** dans AWS CLI

- Pour le désactiver `IPv6` dans l'agent EKS Pod Identity, ajoutez la configuration suivante aux valeurs de configuration du module complémentaire EKS.

Exécutez la AWS CLI commande suivante. Remplacez `my-cluster` par le nom de votre cluster et l'ARN du rôle IAM par le rôle que vous utilisez.

```
aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-
agent \
    --resolve-conflicts PRESERVE --configuration-values '{"agent":
{"additionalArgs": { "-b": "169.254.170.23"}}}'
```

Cette configuration définit l'`IPv4`adresse comme étant la seule adresse utilisée par l'agent.

Amazon EKS applique les modifications aux modules complémentaires EKS en déployant l'agent d'identité Kubernetes DaemonSet pour EKS Pod. Vous pouvez suivre l'état du déploiement dans l'historique des mises à jour du module complémentaire dans AWS Management Console et avec`kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`.

`kubectl rollout`possède les commandes suivantes :

```
kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Si le déploiement prend trop de temps, Amazon EKS annulera le déploiement et un message indiquant le type de mise à jour de l'extension et le statut Echec sera ajouté à l'historique des mises à jour de l'extension. Pour étudier les problèmes éventuels, commencez par l'historique du déploiement et exécutez-le `kubectl logs` sur un pod EKS Pod Identity Agent pour consulter les journaux d'EKS Pod Identity Agent.

Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM avec EKS Pod Identity

Cette rubrique explique comment configurer un compte de Kubernetes service pour qu'il assume un rôle AWS Identity and Access Management (IAM) avec EKS Pod Identity. Tous Pods ceux qui sont configurés pour utiliser le compte de service peuvent alors accéder à tous Service AWS ceux auxquels le rôle est autorisé à accéder.

Pour créer une association EKS Pod Identity, il n'y a qu'une seule étape : vous créez l'association dans AWS EKS via les SDK AWS CloudFormation et d'autres outils. AWS Management Console AWS CLI Il n'y a pas de données ou de métadonnées sur les associations à l'intérieur du cluster dans les objets Kubernetes et vous n'ajoutez pas d'annotations aux comptes de service.

Prérequis

- Un cluster existant. Si vous n'en avez pas, vous pouvez en créer un en suivant l'un des guides Démarrer avec Amazon EKS.
- Le principal IAM qui crée l'association doit avoir le `iam:PassRole`.
- La dernière version AWS CLI installée et configurée sur votre appareil ou AWS CloudShell. Vous pouvez vérifier votre version actuelle avec `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, reportez-vous à la section Installation, mise à jour et désinstallation de la AWS CLI configuration rapide `aws configure` dans le guide de l' AWS Command Line Interface utilisateur. La AWS CLI version installée dans le AWS CloudShell peut également être en retard de plusieurs versions par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Un fichier existant `kubectl config` qui contient la configuration de votre cluster. Pour créer un fichier `kubectl config`, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

Création de l'association d'identité du pod EKS

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire l'agent d'identité du pod EKS.
3. Choisissez l'onglet Access.
4. Dans Associations d'identité du pod, sélectionnez Créer.
5. Pour Rôle IAM, sélectionnez le rôle IAM avec les autorisations dont a besoin la charge de travail.

> **Note**
>
> La liste ne contient que les rôles qui ont la politique d'approbation suivante qui permet à l'identité du pod EKS de les utiliser.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

L'identité du pod EKS utilise `AssumeRole` pour assumer le rôle IAM avant de transmettre les informations d'identification temporaires à vos pods.

`sts:TagSession`

L'identité du pod EKS utilise `TagSession` pour inclure des balises de session dans les demandes adressées à AWS STS.

Vous pouvez utiliser ces balises dans les condition keys de la politique d'approbation pour restreindre les comptes de service, les espaces de noms et les clusters qui peuvent utiliser ce rôle.

Pour obtenir la liste des clés de condition Amazon EKS, consultez la rubrique Conditions définies par Amazon Elastic Kubernetes Service de la Référence de l'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service.

6. Pour Espace de noms Kubernetes, sélectionnez l'espace de noms Kubernetes qui contient le compte de service et la charge de travail. En option, vous pouvez spécifier un espace de noms par nom qui n'existe pas dans le cluster.
7. Pour Compte de service Kubernetes, sélectionnez le compte de service Kubernetes à utiliser. Le manifeste de votre charge de travail Kubernetes doit spécifier ce compte de service. En option, vous pouvez spécifier un compte de service par nom qui n'existe pas dans le cluster.
8. (Facultatif) Pour Balises, choisissez Ajouter une balise pour ajouter des métadonnées dans une paire clé et valeur. Ces balises sont appliquées à l'association et peuvent être utilisées dans les politiques IAM.

   Vous pouvez répéter cette étape pour ajouter plusieurs balises.
9. Choisissez Créer.

AWS CLI

1. Si vous souhaitez associer une politique IAM existante à votre rôle IAM, passez à l'étape suivante.

   Créez une politique IAM. Vous pouvez créer votre propre politique ou copier une politique gérée par AWS , qui accorde certaines des autorisations dont vous avez besoin et la

personnalise en fonction de vos besoins spécifiques. Pour plus d'informations, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

a. Créez un fichier qui inclut les autorisations pour Services AWS , auxquelles vos Pods puissent accéder. Pour obtenir la liste de toutes les actions pour tous Services AWS, consultez la référence d'autorisation de service.

   Vous pouvez exécuter la commande suivante pour créer un exemple de fichier de politique autorisant l'accès en lecture seule à un compartiment Amazon S3. Vous pouvez choisir de stocker les informations de configuration ou un script d'amorçage dans ce compartiment, et les conteneurs de votre Podpeuvent lire le fichier à partir du compartiment et le charger dans votre application. Si vous souhaitez créer cet exemple de politique, copiez le contenu suivant sur votre appareil. Remplacez *my-pod-secrets-bucket* par le nom de votre compartiment et exécutez la commande.

```
cat >my-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. Créez la politique IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. Créez un rôle IAM et associez-le à un compte de service Kubernetes.

   1. Si vous avez un compte de service Kubernetes existant pour lequel vous souhaitez endosser un rôle IAM, vous pouvez ignorer cette étape.

      Création d'un compte de service Kubernetes. Copiez les contenus suivants sur votre appareil. Remplacez *my-service-account* par le nom de votre choix et *default* par

un espace de noms différent, si nécessaire. Si vous changez *default*, l'espace de noms doit déjà exister.

```
cat >my-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

Exécutez la commande suivante.

```
kubectl apply -f my-service-account.yaml
```

2. Exécutez la commande suivante pour créer un fichier de stratégie d'approbation pour le rôle IAM.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
EOF
```

3. Créez le rôle. Remplacez *my-role* avec un nom pour votre rôle IAM, et *my-role-description* avec une description de votre rôle.

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

4. Liez une politique IAM à votre rôle. Remplacez *my-role* par le nom de votre rôle IAM et *my-policy* par le nom d'une politique existante que vous avez créée.

```
aws iam attach-role-policy --role-name my-role --policy-
arn=arn:aws:iam::111122223333:policy/my-policy
```

> **Note**
>
> Contrairement aux rôles IAM des comptes de service, l'identité du pod EKS n'utilise pas d'annotation sur le compte de service.

5. Exécutez la commande suivante pour créer l'association. Remplacez `my-cluster` par le nom du cluster, remplacez *my-service-account* par le nom de votre choix et *par défaut* par un espace de noms différent, si nécessaire.

```
aws eks create-pod-identity-association --cluster-name my-cluster --role-
arn arn:aws:iam::111122223333:role/my-role --namespace default --service-
account my-service-account
```

L'exemple qui suit illustre un résultat.

```
{
    "association": {
        "clusterName": "my-cluster",
        "namespace": "default",
        "serviceAccount": "my-service-account",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws::111122223333:podidentityassociation/my-
cluster/a-abcdefghijklmnop1",
        "associationId": "a-abcdefghijklmnop1",
        "tags": {},
        "createdAt": 1700862734.922,
        "modifiedAt": 1700862734.922
    }
}
```

> **Note**
>
> Vous pouvez spécifier un espace de noms et un compte de service par nom qui n'existe pas dans le cluster. Vous devez créer l'espace de noms, le compte de service et la charge de travail qui utilise le compte de service pour que l'association de l'identité du pod EKS fonctionne.

3. Vérifiez que le rôle et le compte de service sont correctement configurés.

   a. Vérifiez que la stratégie d'approbation du rôle IAM est correctement configurée.

   ```
   aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
   ```

   L'exemple qui suit illustre un résultat.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "Allow EKS Auth service to assume this role for Pod
    Identities",
               "Effect": "Allow",
               "Principal": {
                   "Service": "pods.eks.amazonaws.com"
               },
               "Action": [
                   "sts:AssumeRole",
                   "sts:TagSession"
               ]
           }
       ]
   }
   ```

   b. Vérifiez que la stratégie que vous avez associée à votre rôle lors d'une étape précédente est associée au rôle.

   ```
   aws iam list-attached-role-policies --role-name my-role --query
    AttachedPolicies[].PolicyArn --output text
   ```

   L'exemple qui suit illustre un résultat.

```
arn:aws:iam::111122223333:policy/my-policy
```

c. Définissez une variable pour stocker l'Amazon Resource Name (ARN) de la stratégie que vous souhaitez utiliser. Remplacez *my-policy* par le nom de la stratégie pour laquelle vous voulez confirmer les autorisations.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. Affichez la version par défaut de la stratégie.

```
aws iam get-policy --policy-arn $policy_arn
```

L'exemple qui suit illustre un résultat.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. Consultez le contenu de la stratégie pour vous assurer que la stratégie inclut toutes les autorisations que nécessite votre Pod. Si nécessaire, remplacez *1* dans la commande suivante par la version indiquée dans la sortie précédente.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

L'exemple qui suit illustre un résultat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
```

```
        }
    ]
}
```

Si vous avez créé l'exemple de stratégie lors d'une étape précédente, le résultat est le même. Si vous avez créé une stratégie différente, alors le contenu *example* (exemple) est différent.

Étape suivante

Configurer Pods pour utiliser un compte Kubernetes de service

Configurer Pods pour utiliser un compte Kubernetes de service

Si un Pod a besoin d'y accéder Services AWS, vous devez le configurer pour utiliser un compte Kubernetes de service. Le compte de service doit être associé à un rôle AWS Identity and Access Management (IAM) autorisé à accéder au Services AWS.

Prérequis

- Un cluster existant. Si vous n'en avez pas, vous pouvez en créer un en utilisant l'un des guides Démarrer avec Amazon EKS.
- Un compte de service Kubernetes existant et une association d'identité du pod EKS qui associe le compte de service à un rôle IAM. Le rôle doit être associé à une politique IAM contenant les autorisations que vous souhaitez pour vos Pods qui utilisent Services AWS. Pour plus d'informations sur la façon de créer et de configurer le compte de service et le rôle, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM avec EKS Pod Identity.
- La dernière version AWS CLI installée et configurée sur votre appareil ou AWS CloudShell. Vous pouvez vérifier votre version actuelle avec `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, reportez-vous à la section Installation, mise à jour et désinstallation de la AWS CLI configuration rapide `aws configure` dans le guide de l' AWS Command Line Interface utilisateur. La AWS CLI version installée dans le AWS CloudShell peut également être en retard de plusieurs versions par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version

mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.

- Un fichier existant `kubectl config` qui contient la configuration de votre cluster. Pour créer un fichier `kubectl config`, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

Pour configurer un Pod afin qu'il utilise un compte de service

1. Utilisez la commande suivante pour créer un manifeste de déploiement que vous pouvez déployer pour confirmer la configuration avec un Pod. Remplacez les *`example values`* par vos propres valeurs.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Appliquez le manifeste à votre cluster.

```
kubectl apply -f my-deployment.yaml
```

3. Vérifiez que les variables d'environnement requises existent pour votre Pod.

   a. Consultez les Pods qui ont été déployés lors du déploiement à l'étape précédente.

```
kubectl get pods | grep my-app
```

L'exemple qui suit illustre un résultat.

```
my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
```

b. Vérifiez que le Pod dispose d'un montage de fichier de jetons de compte de service.

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
 AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:
```

L'exemple qui suit illustre un résultat.

```
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:  /var/run/secrets/
pods.eks.amazonaws.com/serviceaccount/eks-pod-identity-token
```

4. Vérifiez que vous Pods pouvez interagir avec le Services AWS en utilisant les autorisations que vous avez attribuées dans la politique IAM associée à votre rôle.

> **Note**
>
> Lorsqu'un Pod utilise les AWS informations d'identification d'un rôle IAM associé à un compte de service, le SDK AWS CLI ou les autres SDK présents dans les conteneurs correspondants Pod utilisent les informations d'identification fournies par ce rôle. Si vous ne restreignez pas l'accès aux informations d'identification fournies au rôle IAM de nœud Amazon EKS, Pod y aura toujours accès. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Si votre Pods ne peut pas interagir avec les services comme prévu, suivez les étapes suivantes pour vérifier que tout est correctement configuré.

a. Vérifiez que vous Pods utilisez une version du AWS SDK qui permet d'assumer un rôle IAM via une association EKS Pod Identity. Pour plus d'informations, consultez Utiliser un AWS SDK compatible.

b. Vérifiez que le déploiement utilise le compte de service.

```
kubectl describe deployment my-app | grep "Service Account"
```

L'exemple qui suit illustre un résultat.

```
Service Account:  my-service-account
```

Définition des autorisations pour que les identités du pod EKS assument des rôles en fonction des balises

L'identité du pod EKS attache des balises aux informations d'identification temporaires de chaque pod avec des attributs tels que le nom du cluster, l'espace de noms, le nom du compte de service. Ces balises de session de rôle permettent aux administrateurs de créer un rôle unique qui peut fonctionner avec tous les comptes de service en autorisant l'accès aux AWS ressources en fonction des balises correspondantes. En ajoutant la prise en charge des balises de session de rôle, les clients peuvent appliquer des frontières de sécurité plus strictes entre les clusters et les charges de travail au sein des clusters, tout en réutilisant les mêmes rôles IAM et politiques IAM.

Par exemple, la politique suivante autorise l'action `s3:GetObject` si l'objet est étiqueté avec le nom du cluster EKS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "s3:ExistingObjectTag/eks-cluster-name": "${aws:PrincipalTag/eks-
cluster-name}"
                }
            }
        }
    ]
}
```

Liste des balises de session ajoutées par l'identité du pod EKS

La liste suivante contient toutes les clés des balises qui sont ajoutées à la demande `AssumeRole` effectuée par Amazon EKS. Pour utiliser ces balises dans les politiques, utilisez `${aws:PrincipalTag/` suivi de la clé, par exemple `${aws:PrincipalTag/kubernetes-namespace}`.

- `eks-cluster-arn`
- `eks-cluster-name`
- `kubernetes-namespace`
- `kubernetes-service-account`
- `kubernetes-pod-name`
- `kubernetes-pod-uid`

Balises inter-comptes

Toutes les balises de session ajoutées par l'identité du pod EKS sont transitives ; les clés et les valeurs des balises sont transmises à toutes les actions `AssumeRole` que vos charges de travail utilisent pour changer de rôle dans un autre compte. Vous pouvez utiliser ces balises dans les politiques d'autres comptes pour limiter l'accès dans des scénarios intercomptes. Pour plus d'informations, consultez Chaînage des rôles avec des balises de session dans le Guide de l'utilisateur IAM.

Balises personnalisées

L'identité du pod EKS ne peut pas ajouter de balises personnalisées supplémentaires à l'action `AssumeRole` qu'elle effectue. Cependant, les balises que vous appliquez au rôle IAM sont toujours disponibles sous le même format : `${aws:PrincipalTag/` suivi de la clé, par exemple `${aws:PrincipalTag/MyCustomTag}`.

> **Note**
>
> Les balises ajoutées à la session via la demande `sts:AssumeRole` sont prioritaires en cas de conflit. Par exemple, supposons qu'Amazon EKS ajoute une clé `eks-cluster-name` et une valeur `my-cluster` à la session lorsque EKS assume le rôle de client. Vous avez également ajouté une balise `eks-cluster-name` au rôle IAM avec la valeur `my-own-cluster`. Dans ce cas, le premier a la priorité et la valeur de la balise `eks-cluster-name` sera `my-cluster`.

Utiliser un AWS SDK compatible

> **Important**
>
> Une version antérieure de la documentation était incorrecte. Le AWS SDK pour Java v1 ne prend pas en charge EKS Pod Identity.

Lors de leur utilisationIdentités du pod EKS, les conteneurs qu'ils contiennent Pods doivent utiliser une version du AWS SDK qui permet d'assumer un rôle IAM depuis l'agent EKS Pod Identity. Assurez-vous que vous utilisez les versions suivantes, ou des versions ultérieures, pour votre AWS SDK :

- Java (version 2) : 2.21.30
- Go v1 : v1.47.11
- Go v2 : release-2023-11-14
- Python (Boto3) — 1.34.41
- Python (botocore) — 1.34.41
- AWS CLI — 1,30,0

  AWS CLI — 2,15,0
- JavaScript v2 — 2,1550,0
- JavaScript v3 — v3.458.0
- Kotlin — v1.0.1
- Ruby : 3.188.0

- Rust — sortie-2024-03-13
- C++ — 1.11.263
- .NET — 3.7.734.0
- PowerShell — 4,1502
- PHP : 3.287.1

Pour vous assurer d'utiliser un kit SDK pris en charge, suivez les instructions d'installation relatives à votre SDK préféré de la page Tools to Build on AWS lorsque vous créez vos conteneurs.

Pour obtenir la liste des modules complémentaires compatibles avec EKS Pod Identity, consultezVersions complémentaires compatibles avec EKS Pod Identity.

Utilisation des informations d'identification de l'identité du pod EKS

Pour utiliser les informations d'identification issues d'une association EKS Pod Identity, votre code peut utiliser n'importe quel AWS SDK pour créer un client pour un AWS service doté d'un SDK. Par défaut, le SDK recherche les informations d' AWS Identity and Access Management identification à utiliser dans une chaîne d'emplacements dans une chaîne d'emplacements. Les informations d'identification de l'identité du pod EKS seront utilisées si vous ne spécifiez pas de fournisseur d'informations d'identification lorsque vous créez le client ou initialisez le kit SDK d'une autre manière.

Cela fonctionne parce que les identités du pod EKS ont été ajoutées au fournisseur d'informations d'identification du conteneur qui est recherché dans une étape de la chaîne d'informations d'identification par défaut. Si vos charges de travail utilisent actuellement des informations d'identification antérieures dans la chaîne de fournisseurs d'informations d'identification, ces informations continueront d'être utilisées même si vous configurez une association d'identité du pod EKS pour la même charge de travail.

Pour plus d'informations sur le fonctionnement des identités du pod EKS, consultez Fonctionnement de l'identité du pod EKS.

Rôle de l'identité du pod EKS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

L'identité du pod EKS utilise `AssumeRole` pour assumer le rôle IAM avant de transmettre les informations d'identification temporaires à vos pods.

`sts:TagSession`

L'identité du pod EKS utilise `TagSession` pour inclure des balises de session dans les demandes adressées à AWS STS.

Vous pouvez utiliser ces balises dans les condition keys de la politique d'approbation pour restreindre les comptes de service, les espaces de noms et les clusters qui peuvent utiliser ce rôle.

Pour obtenir la liste des clés de condition Amazon EKS, consultez la rubrique Conditions définies par Amazon Elastic Kubernetes Service de la Référence de l'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service.

Rôles IAM pour les comptes de service

Les applications situées dans les conteneurs Pod d'un peuvent utiliser un AWS SDK ou le AWS CLI pour envoyer des demandes d'API aux autorisations Services AWS Using AWS Identity and Access Management (IAM). Les applications doivent signer leurs demandes AWS d'API avec des AWS informations d'identification. Les rôles IAM pour les comptes de service offrent une capacité de gestion des informations d'identification à utiliser pour les applications, de la même façon que les profils d'instance Amazon EC2 fournissent des informations d'identification aux instances EC2. Au lieu de créer et de distribuer vos AWS informations d'identification aux conteneurs ou d'utiliser le rôle

de l'instance Amazon EC2, vous associez un rôle IAM à un compte de Kubernetes service et vous le configurez Pods pour utiliser le compte de service. Vous ne pouvez pas utiliser les rôles IAM pour les comptes de service avec les clusters locaux pour Amazon EKS sur AWS Outposts.

Les rôles IAM pour les comptes de service offrent les bénéfices suivants :

- Moindre privilège – Vous pouvez définir les autorisations IAM sur un compte de service et seuls les Pods qui utilisent ce compte de service ont accès à ces autorisations. Cette fonctionnalité élimine également le besoin de solutions tierces telles que `kiam` ou `kube2iam`.
- Isolement des informations d'identification : les conteneurs d'un Pod's peuvent uniquement récupérer les informations d'identification du rôle IAM associées au compte de service que le conteneur utilise. Un conteneur n'a jamais accès aux informations d'identification qui sont utilisées par d'autres conteneurs dans d'autres Pods. Lorsque vous utilisez des rôles IAM pour les comptes de service, les conteneurs de Pod's possèdent également les autorisations attribuées au rôle IAM de nœud Amazon EKS, sauf si vous empêchez les Pod d'accéder au service de métadonnées d'instance (IMDS) Amazon EC2. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.
- Auditabilité — La journalisation des accès et des événements est disponible AWS CloudTrail pour garantir un audit rétrospectif.

Activez les rôles IAM pour les comptes de service en suivant les procédures suivantes :

1. Créez un OIDC fournisseur IAM pour votre cluster — Vous n'effectuez cette procédure qu'une seule fois pour chaque cluster.

> **Note**
>
> Si vous activez le point de terminaison VPC EKS, le point de terminaison du service OIDC EKS ne sera pas accessible depuis ce VPC. Par conséquent, les opérations telles que la création d'un fournisseur OIDC avec `eksctl` dans le VPC n'aboutiront pas et entraîneront une expiration lors de la tentative de demande de `https://oidc.eks.region.amazonaws.com`. Voici un exemple de message d'erreur :
>
> ```
> ** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
> ```
>
> Pour terminer cette étape, vous pouvez exécuter la commande en dehors du VPC, par exemple dans AWS CloudShell ou sur un ordinateur connecté à Internet.

2. Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM — Suivez cette procédure pour chaque ensemble unique d'autorisations que vous souhaitez attribuer à une application.
3. Configurer Pods pour utiliser un compte Kubernetes de service— Complétez cette procédure pour tous ceux Pod qui ont besoin d'accéder à Services AWS.
4. Utilisation d'un kit SDK AWS pris en charge— Vérifiez que la charge de travail utilise un AWS SDK d'une version prise en charge et qu'elle utilise la chaîne d'identification par défaut.

Informations générales IAM, Kubernetes, et OpenID Connect (OIDC)

En 2014, AWS Identity and Access Management ajout de la prise en charge des identités fédérées à l'aide de OpenID Connect (OIDC). Cette fonctionnalité vous permet d'authentifier les appels d' AWS API auprès des fournisseurs d'identité pris en charge et de recevoir un jeton OIDC JSON Web valide (JWT). Vous pouvez transmettre ce jeton à l'opération AWS STS `AssumeRoleWithWebIdentity` API et recevoir des informations d'identification de rôle temporaires IAM. Vous pouvez utiliser ces informations d'identification pour interagir avec n'importe quel Service AWS, y compris Amazon S3 et DynamoDB.

Chaque jeton JWT est signé par une paire de clés de signature. Les clés sont servies sur le fournisseur OIDC géré par Amazon EKS et la clé privée change tous les 7 jours. Amazon EKS conserve les clés publiques jusqu'à leur expiration. Si vous connectez des clients OIDC externes, sachez que vous devez actualiser les clés de signature avant l'expiration de la clé publique. Découvrez comment the section called “Récupérez les clés de signature”.

Kubernetes utilise depuis longtemps les comptes de service comme son propre système d'identité interne. Pods peut s'authentifier auprès du serveur API Kubernetes utilisant un jeton monté automatiquement (qui était un JWT non-OIDC) que seul le serveur API Kubernetes a pu valider. Ces jetons de compte de service hérités n'expirent pas et la rotation de la clé de signature est un processus difficile. Dans Kubernetes version `1.12`, le support a été ajouté pour une nouvelle fonction `ProjectedServiceAccountToken`. Cette fonctionnalité est un jeton Web OIDC JSON qui contient également l'identité du compte de service et prend en charge une audience configurable.

Amazon EKS héberge un point de terminaison de découverte OIDC public pour chaque cluster contenant les clés de signature des jetons web `ProjectedServiceAccountToken` JSON afin que les systèmes externes, comme IAM, puissent valider et accepter les jetons OIDC émis par Kubernetes.

Créez un OIDC fournisseur IAM pour votre cluster

Votre cluster possède une URL de diffuseur OpenID Connect (OIDC) qui lui est associée. Pour utiliser des rôles AWS Identity and Access Management (IAM) pour les comptes de service, un OIDC fournisseur IAM doit exister pour l'URL de l'OIDCémetteur de votre cluster.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Un fichier existant `kubectl config` qui contient la configuration de votre cluster. Pour créer un fichier `kubectl config`, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

Vous pouvez créer un fournisseur OIDC IAM pour votre cluster en utilisant `eksctl` ou la AWS Management Console.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour créer un fournisseur d'identité OIDC IAM pour votre cluster avec **`eksctl`**

1. Déterminez l'identifiant de l'émetteur OIDC pour votre cluster.

   Récupérez l'identifiant de l'émetteur OIDC de votre cluster et stockez-le dans une variable. Remplacez *`my-cluster`* par votre propre valeur.

   ```
   cluster_name=my-cluster
   ```

   ```
   oidc_id=$(aws eks describe-cluster --name $cluster_name --query
    "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
   ```

   ```
   echo $oidc_id
   ```

2. Déterminez si un fournisseur OIDC IAM avec l'identifiant de l'émetteur de votre cluster est déjà présent sur votre compte.

   ```
   aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
   ```

   Si la sortie est renvoyée, cela signifie que vous disposez déjà d'un fournisseur OIDC IAM pour votre cluster. Vous pouvez donc ignorer l'étape suivante. Si aucune sortie n'est renvoyée, vous devez créer un fournisseur OIDC IAM pour votre cluster.

3. Créez votre fournisseur d'identité OIDC IAM pour votre cluster avec la commande suivante.

   ```
   eksctl utils associate-iam-oidc-provider --cluster $cluster_name --approve
   ```

   > **Note**
   >
   > Si vous activez le point de terminaison VPC EKS, le point de terminaison du service OIDC EKS ne sera pas accessible depuis ce VPC. Par conséquent, les opérations telles que la création d'un fournisseur OIDC avec `eksctl` dans le VPC n'aboutiront pas et entraîneront une expiration lors de la tentative de demande de `https://oidc.eks.`*`region`*`.amazonaws.com`. Voici un exemple de message d'erreur :
   >
   > ```
   > ** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
   > ```

Pour terminer cette étape, vous pouvez exécuter la commande en dehors du VPC, par exemple dans AWS CloudShell ou sur un ordinateur connecté à Internet.

AWS Management Console

Pour créer un fournisseur d'OIDCidentité IAM pour votre cluster à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le volet de gauche, sélectionnez Clusters, puis sélectionnez le nom de votre cluster sur la page Clusters.
3. Dans la section Détails de l'onglet Présentation, notez la valeur de l'URL du fournisseur d'OpenID Connect.
4. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
5. Dans le panneau de navigation, sélectionnez Identity Providers (Fournisseurs d'identité) sous Access management (Gestion des accès). Si un fournisseur est répertorié et correspond à l'URL de votre cluster, alors vous avez déjà un fournisseur pour votre cluster. Si aucun fournisseur ne correspond à l'URL de votre cluster, vous devez en créer un.
6. Pour créer un fournisseur, cliquez sur Ajouter un fournisseur.
7. Pour Provider type (Type de fournisseur), sélectionnez OpenID Connect.
8. Pour l'URL du fournisseur, entrez l'URL du fournisseur OIDC pour votre cluster, puis choisissez Obtenir une empreinte.
9. Pour le Audience (Public ciblé), saisissez **`sts.amazonaws.com`** et choisissez Add provider (Ajouter un fournisseur).

Étape suivante

Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM

Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM

Cette rubrique explique comment configurer un compte de Kubernetes service pour qu'il assume un rôle AWS Identity and Access Management (IAM). Tous les Pods configurés pour utiliser le compte

de service peuvent ensuite accéder à n'importe quel Service AWS auxquels le rôle est autorisé à accéder.

Prérequis

- Un cluster existant. Si vous n'en avez pas, vous pouvez en créer un en suivant l'un des guides Démarrer avec Amazon EKS.
- Un fournisseur IAM OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un ou comment en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Un fichier existant `kubectl config` qui contient la configuration de votre cluster. Pour créer un fichier `kubectl config`, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

Pour associer un rôle IAM à un compte de service Kubernetes

1. Si vous souhaitez associer une politique IAM existante à votre rôle IAM, passez à l'étape suivante.

   Créez une politique IAM. Vous pouvez créer votre propre politique ou copier une politique AWS gérée qui accorde déjà certaines des autorisations dont vous avez besoin et la personnaliser en

fonction de vos besoins spécifiques. Pour plus d'informations, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

a. Créez un fichier qui inclut les autorisations pour le fichier Services AWS auquel vous Pods souhaitez accéder. Pour obtenir la liste de toutes les actions pour tous Services AWS, consultez la référence d'autorisation de service.

   Vous pouvez exécuter la commande suivante pour créer un exemple de fichier de politique autorisant l'accès en lecture seule à un compartiment Amazon S3. Vous pouvez choisir de stocker les informations de configuration ou un script d'amorçage dans ce compartiment, et les conteneurs de votre Podpeuvent lire le fichier à partir du compartiment et le charger dans votre application. Si vous souhaitez créer cet exemple de politique, copiez le contenu suivant sur votre appareil. Remplacez *my-pod-secrets-bucket* par le nom de votre compartiment et exécutez la commande.

```
cat >my-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. Créez la politique IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. Créez un rôle IAM et associez-le à un compte de service Kubernetes. Sinon, vous pouvez utiliser `eksctl` ou AWS CLI.

   eksctl

   Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Remplacez *`my-service-account`* avec le nom du compte de service Kubernetes pour lequel vous souhaitez que `eksctl` crée et associe avec un rôle IAM. Remplacez *`default`* (défaut) par l'espace de noms dans lequel vous souhaitez que `eksctl` créer le compte de service. Remplacez *`my-cluster`* par le nom de votre cluster. Remplacez *`my-role`* par le nom du rôle auquel vous souhaitez associer le compte de service. S'il n'existe pas déjà, `eksctl` le crée pour vous. Remplacez *`111122223333`* par l'ID de votre compte et *`my-policy`* par le nom d'une stratégie existante.

```
eksctl create iamserviceaccount --name my-service-account --namespace default --
cluster my-cluster --role-name my-role \
    --attach-policy-arn arn:aws:iam::111122223333:policy/my-policy --approve
```

> **Important**
>
> Si le rôle ou le compte de service existe déjà, la commande précédente peut échouer. `eksctl` propose différentes options que vous pouvez fournir dans ces situations. Pour de plus amples informations, exécutez **`eksctl create iamserviceaccount --help`**.

AWS CLI

1. Si vous avez un compte de service Kubernetes existant pour lequel vous souhaitez endosser un rôle IAM, vous pouvez ignorer cette étape.

   Création d'un compte de service Kubernetes. Copiez les contenus suivants sur votre appareil. Remplacez *`my-service-account`* par le nom de votre choix et *`default`* par un espace de noms différent, si nécessaire. Si vous changez *`default`*, l'espace de noms doit déjà exister.

   ```
   cat >my-service-account.yaml <<EOF
   apiVersion: v1
   kind: ServiceAccount
   metadata:
   ```

```
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

2. Définissez votre Compte AWS ID sur une variable d'environnement à l'aide de la commande suivante.

```
account_id=$(aws sts get-caller-identity --query "Account" --output text)
```

3. Définissez le fournisseur d'identité OIDC de votre cluster sur une variable d'environnement avec la commande suivante. Remplacez *my-cluster* par le nom de votre cluster.

```
oidc_provider=$(aws eks describe-cluster --name my-cluster --region
 $AWS_REGION --query "cluster.identity.oidc.issuer" --output text | sed -e "s/
^https:\/\///")
```

4. Définissez des variables pour l'espace de noms et le nom du compte de service. Remplacez *my-service-account* par le compte de service Kubernetes auquel vous souhaitez assumer le rôle. Remplacez *default* par l'espace de noms du compte de service.

```
export namespace=default
export service_account=my-service-account
```

5. Exécutez la commande suivante pour créer un fichier de stratégie d'approbation pour le rôle IAM. Si vous souhaitez autoriser tous les comptes de service d'un espace de noms à utiliser le rôle, copiez le contenu suivant sur votre appareil. Remplacez *StringEquals*par **StringLike** et remplacez *$service_account* par. * Vous pouvez ajouter plusieurs entrées dans les conditions `StringEquals` ou `StringLike` pour autoriser plusieurs comptes de service ou espaces de noms à endosser le rôle. Pour autoriser des rôles provenant d'un autre Compte AWS que le compte dans lequel se trouve votre cluster pour assumer le rôle, consultez Autorisations IAM entre comptes pour en savoir plus.

```
cat >trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::$account_id:oidc-provider/$oidc_provider"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "$oidc_provider:aud": "sts.amazonaws.com",
          "$oidc_provider:sub": "system:serviceaccount:
$namespace:$service_account"
        }
      }
    }
  ]
}
EOF
```

6. Créez le rôle. Remplacez *my-role* avec un nom pour votre rôle IAM, et *my-role-description* avec une description de votre rôle.

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

7. Liez une politique IAM à votre rôle. Remplacez *my-role* par le nom de votre rôle IAM et *my-policy* par le nom d'une politique existante que vous avez créée.

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws:iam::
$account_id:policy/my-policy
```

8. Annotez votre compte de service avec l'Amazon Resource Name (ARN) du rôle IAM que vous souhaitez que le compte de service endosse. Remplacez *my-role* par le nom de votre rôle IAM existant. Supposons que vous ayez autorisé un rôle provenant d'un compte différent de Compte AWS celui dans lequel se trouve votre cluster à assumer ce rôle lors d'une étape précédente. Assurez-vous ensuite de spécifier le rôle Compte AWS et à partir de l'autre compte. Pour plus d'informations, consultez Autorisations IAM entre comptes.

```
kubectl annotate serviceaccount -n $namespace $service_account
 eks.amazonaws.com/role-arn=arn:aws:iam::$account_id:role/my-role
```

3. Vérifiez que le rôle et le compte de service sont correctement configurés.

   a. Vérifiez que la stratégie d'approbation du rôle IAM est correctement configurée.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

L'exemple qui suit illustre un résultat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:default:my-
service-account",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
```

b. Vérifiez que la stratégie que vous avez associée à votre rôle lors d'une étape précédente est associée au rôle.

```
aws iam list-attached-role-policies --role-name my-role --query
 AttachedPolicies[].PolicyArn --output text
```

L'exemple qui suit illustre un résultat.

```
arn:aws:iam::111122223333:policy/my-policy
```

c. Définissez une variable pour stocker l'Amazon Resource Name (ARN) de la stratégie que vous souhaitez utiliser. Remplacez *my-policy* par le nom de la stratégie pour laquelle vous voulez confirmer les autorisations.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. Affichez la version par défaut de la stratégie.

```
aws iam get-policy --policy-arn $policy_arn
```

L'exemple qui suit illustre un résultat.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. Consultez le contenu de la stratégie pour vous assurer que la stratégie inclut toutes les autorisations que nécessite votre Pod. Si nécessaire, remplacez *1* dans la commande suivante par la version indiquée dans la sortie précédente.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

L'exemple qui suit illustre un résultat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Si vous avez créé l'exemple de stratégie lors d'une étape précédente, le résultat est le même. Si vous avez créé une stratégie différente, alors le contenu *example* (exemple) est différent.

f. Confirmez que le compte de service Kubernetes est annoté avec le rôle.

```
kubectl describe serviceaccount my-service-account -n default
```

L'exemple qui suit illustre un résultat.

```
Name:                my-service-account
Namespace:           default
Annotations:         eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/my-role
Image pull secrets:  <none>
Mountable secrets:   my-service-account-token-qqjfl
Tokens:              my-service-account-token-qqjfl
[...]
```

4. (Facultatif)Configuration du AWS Security Token Service point de terminaison pour un compte de service. AWS recommande d'utiliser un point de AWS STS terminaison régional plutôt que le point de terminaison global. Cela réduit la latence, fournit une redondance intégrée et augmente la validité des jetons de session.

Étape suivante

Configurer Pods pour utiliser un compte Kubernetes de service

Configurer Pods pour utiliser un compte Kubernetes de service

Si un Pod a besoin d'y accéder Services AWS, vous devez le configurer pour utiliser un compte Kubernetes de service. Le compte de service doit être associé à un rôle AWS Identity and Access Management (IAM) autorisé à accéder au Services AWS.

Prérequis

- Un cluster existant. Si vous n'en avez pas, vous pouvez en créer un en utilisant l'un des guides Démarrer avec Amazon EKS.

- Un fournisseur IAM OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un ou comment en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Un compte de service Kubernetes existant qui est associé à un rôle IAM. Le compte de service doit être annoté avec l'Amazon Resource Name (ARN) du rôle IAM. Le rôle doit être associé à une politique IAM contenant les autorisations que vous souhaitez pour vos Pods qui utilisent Services AWS. Pour plus d'informations sur la façon de créer et de configurer le compte de service et le rôle, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.
- Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Un fichier existant `kubectl config` qui contient la configuration de votre cluster. Pour créer un fichier `kubectl config`, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

Pour configurer un Pod afin qu'il utilise un compte de service

1. Utilisez la commande suivante pour créer un manifeste de déploiement que vous pouvez déployer pour confirmer la configuration avec un Pod. Remplacez les *`example values`* par vos propres valeurs.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
```

```
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Appliquez le manifeste à votre cluster.

```
kubectl apply -f my-deployment.yaml
```

3. Vérifiez que les variables d'environnement requises existent pour votre Pod.

   a. Consultez les Pods qui ont été déployés lors du déploiement à l'étape précédente.

   ```
   kubectl get pods | grep my-app
   ```

   L'exemple qui suit illustre un résultat.

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Consultez l'ARN du rôle IAM qu'utilise Pod.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_ROLE_ARN:
   ```

   L'exemple qui suit illustre un résultat.

   ```
   AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-role
   ```

L'ARN du rôle doit correspondre à l'ARN du rôle avec lequel vous avez annoté le compte de service existant. Pour en savoir plus sur l'annotation du compte de service, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

c. Confirmez que Pod possède un fichier de jeton d'identité Web pour un montage.

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
 AWS_WEB_IDENTITY_TOKEN_FILE:
```

L'exemple qui suit illustre un résultat.

```
AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

Le `kubelet` demande et stocke le jeton au nom du Pod. Par défaut, `kubelet` actualise le jeton si ce dernier est supérieur à 80 % de son temps total avant le live, ou si le jeton est âgé de plus de 24 heures. Vous pouvez modifier la durée d'expiration de n'importe quel compte autre que le compte de service par défaut en utilisant les paramètres dans votre spécification de Pod. Pour plus d'informations, consultez la section Projection du volume des jetons de compte de service dans la documentation Kubernetes.

Le Webhook d'identité de pod Amazon EKS sur le cluster surveille les Pods qui utilisent un compte de service avec l'annotation suivante :

```
eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-role
```

Le webhook applique les variables d'environnement précédentes à ces Pods. Votre cluster n'a pas besoin d'utiliser le webhook pour configurer les variables d'environnement et les montages de fichiers de jetons. Vous pouvez configurer manuellement Pods pour obtenir ces variables d'environnement. Les versions prises en charge du AWS SDK recherchent d'abord ces variables d'environnement dans le fournisseur de chaîne d'informations d'identification. Les informations d'identification du rôle sont utilisées pour les Pods qui répondent à ces critères.

4. Vérifiez que vous Pods pouvez interagir avec le Services AWS en utilisant les autorisations que vous avez attribuées dans la politique IAM associée à votre rôle.

> **Note**
>
> Lorsqu'un Pod utilise les AWS informations d'identification d'un rôle IAM associé à un compte de service, le SDK AWS CLI ou les autres SDK présents dans les conteneurs correspondants Pod utilisent les informations d'identification fournies par ce rôle. Si vous ne restreignez pas l'accès aux informations d'identification fournies au rôle IAM de nœud Amazon EKS, Pod y aura toujours accès. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Si votre Pods ne peut pas interagir avec les services comme prévu, suivez les étapes suivantes pour vérifier que tout est correctement configuré.

a. Vérifiez que vous Pods utilisez une version du AWS SDK qui prend en charge le rôle IAM via un fichier de jeton d'identité OpenID Connect Web. Pour plus d'informations, consultez Utilisation d'un kit SDK AWS pris en charge.

b. Vérifiez que le déploiement utilise le compte de service.

```
kubectl describe deployment my-app | grep "Service Account"
```

L'exemple qui suit illustre un résultat.

```
Service Account:  my-service-account
```

c. Si votre Pods ne parvient toujours pas à accéder aux services, consultez les étapes qui sont décrites dans Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM pour vérifier que votre rôle et votre compte de service sont correctement configurés.

Configuration du AWS Security Token Service point de terminaison pour un compte de service

Si vous utilisez un compte de Kubernetes service avecRôles IAM pour les comptes de service, vous pouvez configurer le type de AWS Security Token Service point de terminaison utilisé par le compte de service si la version de votre cluster et de votre plateforme est identique ou ultérieure à celles répertoriées dans le tableau suivant. Si la version de votre Kubernetes ou celle de votre plateforme

est antérieure à celles répertoriées dans le tableau, alors vos comptes de service ne peuvent utiliser que le point de terminaison international.

Version de Kubernetes	Version de la plateforme	Type de point de terminaison par défaut
1.30	eks.2	Régional
1.29	eks.1	Régional
1.28	eks.1	Régional
1.27	eks.1	Régional
1.26	eks.1	Régional
1.25	eks.1	Régional
1.24	eks.2	Régional
1.23	eks.1	Régional

AWS recommande d'utiliser les AWS STS points de terminaison régionaux plutôt que le point de terminaison global. Cela réduit la latence, fournit une redondance intégrée et augmente la validité des jetons de session. Le AWS Security Token Service doit être actif Région AWS là où il Pod s'exécute. De plus, votre application doit disposer d'une redondance intégrée pour une Région AWS solution différente en cas de panne du service dans le. Région AWS Pour plus d'informations, consultez la section Gestion AWS STS dans et Région AWS dans le guide de l'utilisateur IAM.

Prérequis

- Un cluster existant. Si vous n'en avez pas, vous pouvez en créer un en utilisant l'un des guides Démarrer avec Amazon EKS.
- Un fournisseur IAM OIDC existant pour votre cluster. Pour plus d'informations, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Un compte de service Kubernetes existant configuré pour être utilisé avec la fonction Amazon EKS IAM pour les comptes de service.

Pour configurer le type de point de terminaison utilisé par un compte de service Kubernetes

Les exemples suivants utilisent tous le compte de service `aws-node` de Kubernetesutilisé par le plugin CNI Amazon VPC. Vous pouvez remplacer les *`example values`* par vos propres comptes de service, Pods, espaces de noms et autres ressources.

1. Sélectionnez un Pod qui utilise un compte de service dont vous souhaitez modifier le point de terminaison. Déterminez dans Région AWS lequel il Pod se trouve. Remplacez *`aws-node-6mfgv`* par votre nom de Pod et *`kube-system`* par l'espace de nom de votre Pod.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep Node:
   ```

   L'exemple qui suit illustre un résultat.

   ```
   ip-192-168-79-166.us-west-2/192.168.79.166
   ```

   Région AWS
2. Déterminez le type de point de terminaison utilisé par le compte de service du Pod's.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
   ```

   L'exemple qui suit illustre un résultat.

   ```
   AWS_STS_REGIONAL_ENDPOINTS: regional
   ```

   Si le point de terminaison actuel est international, alors `global` est renvoyé en sortie. Si aucune sortie n'est renvoyée, le type de point de terminaison par défaut est en cours d'utilisation et n'a pas été remplacé.
3. Si la version de votre cluster ou celle de votre plateforme est identique ou ultérieure à celles répertoriées dans le tableau, vous pouvez modifier le type de point de terminaison utilisé par votre compte de service pour le faire passer du type par défaut à un type différent à l'aide de l'une des commandes suivantes. Remplacez *`aws-node`* par le nom de votre compte de service et *`kube-system`* par l'espace de noms de votre compte de service.

   - Si votre type de point de terminaison par défaut ou actuel est international et que vous souhaitez le remplacer par régional :

```
kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=true
```

Si vous utilisez Rôles IAM pour les comptes de service pour générer des URL S3 présignées dans votre application exécutée dans des conteneurs de Pods, le format de l'URL des points de terminaison régionaux est similaire à l'exemple suivant :

```
https://bucket.s3.us-west-2.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
```

- Si votre type de point de terminaison par défaut ou actuel est régional et que vous souhaitez le remplacer par international :

```
kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=false
```

Si votre application envoie explicitement des demandes à des points de terminaison AWS STS globaux et que vous ne remplacez pas le comportement par défaut consistant à utiliser des points de terminaison régionaux dans les clusters Amazon EKS, les demandes échoueront avec une erreur. Pour plus d'informations, consultez Les conteneurs Pod reçoivent l'erreur suivante : `An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`.

Si vous utilisez Rôles IAM pour les comptes de service pour générer des URL S3 présignées dans votre application exécutée dans des conteneurs de Pods, le format de l'URL des points de terminaison internationaux est similaire à l'exemple suivant :

```
https://bucket.s3.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
```

Si vous disposez d'une automatisation qui attend l'URL pré-signée dans un certain format ou si votre application ou les dépendances en aval qui utilisent des URL pré-signées ont des attentes à l'égard de l'URL Région AWS cible, apportez les modifications nécessaires pour utiliser le point de terminaison approprié. AWS STS

4. Supprimez et recréez tous les Pods existants associés au compte de service pour appliquer les variables d'environnement d'informations d'identification. Le hook web en mutation ne les applique pas aux Pods qui sont déjà en cours d'exécution. Vous pouvez remplacer *Pods*, *kube-system* et *-l k8s-app=aws-node* par les informations sur des Pods pour lesquels vous avez défini votre annotation.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

5. Vérifiez que les Pods ont tous redémarré.

```
kubectl get Pods -n kube-system -l k8s-app=aws-node
```

6. Affichez les variables d'environnement pour l'un des Pods. Vérifiez que la valeur `AWS_STS_REGIONAL_ENDPOINTS` est celle que vous avez définie à une étape précédente.

```
kubectl describe pod aws-node-kzbtr -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

L'exemple qui suit illustre un résultat.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Autorisations IAM entre comptes

Vous pouvez configurer les autorisations IAM entre comptes en créant un fournisseur d'identité à partir du cluster d'un autre compte ou en utilisant les opérations `AssumeRole` chaînées. Dans les exemples suivants, Account A (compte A) possède un cluster Amazon EKS qui prend en charge les rôles IAM pour les comptes de service. Les Pods exécutés sur ce cluster doivent endosser les autorisations IAM de Account B (compte B).

Example Créer un fournisseur d'identité à partir du cluster d'un autre compte

Example

Dans cet exemple, le Compte A fournit au Compte B l'URL de l'émetteur OpenID Connect (OIDC) à partir de son cluster. Le compte B suit les instructions de Créez un OIDC fournisseur IAM pour votre cluster et Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM avec l'URL de l'émetteur OIDC du cluster du compte A. Ensuite, un administrateur de cluster annote le compte de service du cluster du compte A pour utiliser le rôle du compte B (*444455556666*).

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::444455556666:role/account-b-role
```

Example Utiliser des opérations **`AssumeRole`** chaînées

Example

Dans cet exemple, le compte B crée une politique IAM avec les autorisations à accorder aux Pods du cluster du compte A. Le compte B (*444455556666*) attache cette politique à un rôle IAM avec une relation d'approbation qui offre les autorisations `AssumeRole` au compte A (*111122223333*), comme illustré ci-dessous.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Le compte A crée un rôle avec une politique d'approbation qui récupère les informations d'identification du fournisseur d'identité créé avec l'adresse de l'émetteur OIDC.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.region-
code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity"
```

```
    }
  ]
}
```

Si votre cluster se trouve dans une autre , ou si vous avez copié les images dans votre propre référentiel lors d'une étape précédente, effectuez les étapes suivantes.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::444455556666:role/account-b-role"
        }
    ]
}
```

Le code d'application pour que les Pods endossent le rôle du compte B utilise deux profils : `account_b_role` et `account_a_role`. Le profil `account_b_role` utilise le profil `account_a_role` comme source. Pour le AWS CLI, le `~/.aws/config` fichier est similaire au suivant.

```
[profile account_b_role]
source_profile = account_a_role
role_arn=arn:aws:iam::444455556666:role/account-b-role

[profile account_a_role]
web_identity_token_file = /var/run/secrets/eks.amazonaws.com/serviceaccount/token
role_arn=arn:aws:iam::111122223333:role/account-a-role
```

Pour spécifier des profils chaînés pour d'autres AWS SDK, consultez la documentation du SDK que vous utilisez. Pour plus d'informations, consultez la section Outils sur lesquels vous pouvez vous appuyer AWS.

Utilisation d'un kit SDK AWS pris en charge

Lors de leur utilisationRôles IAM pour les comptes de service, les conteneurs qu'ils contiennent Pods doivent utiliser une version du AWS SDK qui permet d'assumer un rôle IAM via un fichier de jeton d'identité OpenID Connect Web. Assurez-vous que vous utilisez les versions suivantes, ou des versions ultérieures, pour votre AWS SDK :

- Java (version 2) – 2.10.11
- Java – 1.11.704
- Go – 1.23.13
- Python (Boto3) – 1.9.220
- Python (botocore) – 1.12.200
- AWS CLI — 1,16,232
- Node – 2.525.0 et 3.27.0
- Ruby – 3.58.0
- C++ – 1.7.174
- .NET – 3.3.659.1 – Vous devez également inclure `AWSSDK.SecurityToken`.
- PHP – 3.110.7

De nombreux modules complémentaires Kubernetes populaires, tels que le Cluster Autoscaler, Qu'est-ce que AWS Load Balancer Controller ? et le Amazon VPC CNI plugin for Kubernetes, prennent en charge les rôles IAM pour les comptes de service.

Pour vous assurer d'utiliser un kit SDK pris en charge, suivez les instructions d'installation relatives à votre SDK préféré de la page Tools to Build on AWS lorsque vous créez vos conteneurs.

Utilisation des informations d'identification

Pour utiliser les informations d'identification des rôles IAM pour les comptes de service, votre code peut utiliser n'importe quel AWS SDK pour créer un client pour un AWS service doté d'un SDK. Par défaut, le SDK recherche les informations d'identification à utiliser dans une chaîne d'emplacements dans une chaîne d' AWS Identity and Access Management emplacements. Les informations d'identification des rôles IAM des comptes de service seront utilisées si vous ne spécifiez pas de fournisseur d'informations d'identification lors de la création du client ou de l'initialisation du kit SDK.

Cela fonctionne parce que les rôles IAM des comptes de service ont été ajoutés en tant qu'étape dans la chaîne d'informations d'identification par défaut. Si vos charges de travail utilisent actuellement des informations d'identification qui se trouvent plus haut dans la chaîne d'informations d'identification, ces informations d'identification continueront d'être utilisées même si vous configurez un rôle IAM des comptes de service pour la même charge de travail.

Le SDK échange automatiquement le OIDC jeton du compte de service contre des informations d'identification temporaires à AWS Security Token Service l'aide de

l'`AssumeRoleWithWebIdentity`action. Amazon EKS et cette action du kit SDK poursuivent la rotation des informations d'identification temporaires en les renouvelant avant qu'elles n'expirent.

Récupérez les clés de signature

Kubernetesémet un `ProjectedServiceAccountToken` pour chacun KubernetesService Account. Ce jeton est un OIDC jeton, qui est en outre un type deJSON web token (JWT). Amazon EKS héberge un point de OIDC terminaison public pour chaque cluster qui contient les clés de signature du jeton afin que les systèmes externes puissent le valider.

Pour valider un`ProjectedServiceAccountToken`, vous devez récupérer les clés de signature OIDC publiques, également appelées. JSON Web Key Set (JWKS) Utilisez ces clés dans votre application pour valider le jeton. Par exemple, vous pouvez utiliser la bibliothèque Python PyJWT pour valider des jetons à l'aide de ces clés. Pour plus d'informations sur le`ProjectedServiceAccountToken`, voirthe section called "Informations générales IAM, Kubernetes, et OpenID Connect (OIDC)".

Prérequis

- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- AWS CLI— Un outil de ligne de commande permettant de travailler avec AWS des services, notamment Amazon EKS. Pour plus d'informations, consultez Installation, mise à jour et désinstallation d' AWS CLI dans le Guide de l'utilisateur AWS Command Line Interface . Après l'avoir installé AWS CLI, nous vous recommandons de le configurer également. Pour plus d'informations, consultez Configuration rapide avec `aws configure` dans le Guide de l'utilisateur AWS Command Line Interface .

Récupérer les clés de signature OIDC publiques ()AWS CLI

1. Récupérez l'OIDCURL de votre cluster Amazon EKS à l'aide du AWS CLI.

```
$ aws eks describe-cluster --name my-cluster --query 'cluster.identity.oidc.issuer'
"https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE"
```

2. Récupérez la clé de signature publique à l'aide curl d'un outil similaire ou d'un outil similaire. Le résultat est un JSON Web Key Set (JWKS).

> **Important**
>
> Amazon EKS limite les appels vers le point de terminaison. OIDC Vous devez mettre en cache la clé de signature publique. Respectez l'`cache-control`en-tête inclus dans la réponse.

> **Important**
>
> Amazon EKS fait pivoter la clé OIDC de signature tous les sept jours.

```
$ curl https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE/keys
{"keys":
[{"kty":"RSA","kid":"2284XXXX4a40","use":"sig","alg":"RS256","n":"wklbXXXXMVfQ","e":"AQAB"}
```

Nœuds Amazon EKS

Un nœud Kubernetes est une machine qui exécute des applications conteneurisées. Chaque nœud dispose des composants suivants :

- Environnement d'exécution du conteneur : logiciel permettant d'exécuter les conteneurs.
- **`kubelet`** : vérifie que les conteneurs sont sains et fonctionnent dans le Pod associé.
- **`kube-proxy`** : gère les règles réseau qui permettent la communication avec vos Pods.

Pour plus d'informations, consultez Nœuds dans la documentation Kubernetes.

Votre cluster Amazon EKS peut programmer des Pods sur n'importe quelle combinaison de nœuds autogérés, groupes de nœuds gérés Amazon EKS et AWS Fargate. Pour en savoir plus sur les nœuds déployés dans votre cluster, consultez Afficher les ressources Kubernetes.

Important

AWS Fargate avec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Note

Les nœuds doivent se trouver dans le même VPC que les sous-réseaux que vous avez sélectionnés lorsque vous avez créé le cluster. Toutefois, les nœuds ne doivent pas nécessairement se trouver dans les mêmes sous-réseaux.

Le tableau suivant fournit plusieurs critères à évaluer pour décider quelles options répondent le mieux à vos besoins. Ce tableau n'inclut pas les nœuds connectés qui ont été créés en dehors d'Amazon EKS, qui peuvent seulement être consultés.

> **Note**
>
> Bottlerocket présente quelques différences spécifiques par rapport aux informations générales de ce tableau. Pour plus d'informations, consultez la documentation de Bottlerocket sur GitHub.

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Possibilité d'être déployé dans AWS Outposts	Non	Oui	Non
Possibilité d'être déployé dans une zone locale AWS	Non	Oui : pour plus d'informations, consultez Amazon EKS et AWS Local Zones.	Non
Possibilité d'exécuter des conteneurs qui nécessitent Windows	Oui	Oui : votre cluster nécessite toujours au moins un nœud Linux (deux recommandés pour la disponibilité).	Non
Possibilité d'exécuter des conteneurs qui nécessitent Linux	Oui	Oui	Oui
Possibilité d'exécuter des applications nécessitant la puce Inferentia	Oui : nœuds Amazon Linux uniquement	Oui : amazon Linux uniquement	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Possibilité d'exécuter des applications nécessitant un GPU	Oui : nœuds Amazon Linux uniquement	Oui : amazon Linux uniquement	Non
Possibilité d'exécuter des applications nécessitant des processeurs Arm	Oui	Oui	Non
Peut exécuter AWS Bottlerocket	Oui	Oui	Non
Les pods partagent un environnement d'exécution du noyau avec d'autres Pods	Oui : tous vos Pods sur chacun de vos nœuds	Oui : tous vos Pods sur chacun de vos nœuds	Non : chaque Pod a un noyau dédié
Les pods partagent le processeur, la mémoire, le stockage et les ressources réseau avec d'autres Pods.	Oui : possibilité d'entraîner des ressources inutilisées sur chaque nœud	Oui : possibilité d'entraîner des ressources inutilisées sur chaque nœud	Non : chaque Pod dispose de ressources dédiées et peut être dimensionné indépendamment pour optimiser l'utilisation des ressources.

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Les pods peuvent utiliser plus de matériel et de mémoire que demandé dans les spécifications des Pod	Oui : si le Pod nécessite plus de ressources que demandé et que des ressources sont disponibles sur le nœud, le Pod peut utiliser des ressources supplémentaires.	Oui : si le Pod nécessite plus de ressources que demandé et que des ressources sont disponibles sur le nœud, le Pod peut utiliser des ressources supplémentaires.	Non : le Pod peut toutefois être redéployé à l'aide d'une plus grande configuration de vCPU et de mémoire.
Doit déployer et gérer des instances Amazon EC2	Oui : automatisé via Amazon EKS si vous avez déployé une AMI optimisée pour Amazon EKS. Si vous avez déployé une AMI personnalisée, vous devez alors mettre à jour l'instance manuellement.	Oui : configuration manuelle ou utilisation des modèles AWS CloudFormation fournis par Amazon EKS pour déployer des nœuds Linux (x86), Linux(Arm) ou Windows.	Non
Doit sécuriser, entretenir et corriger le système d'exploitation des instances Amazon EC2	Oui	Oui	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Peut fournir des arguments d'amorçage lors du déploiement d'un nœud, tels que des arguments `kubelet` supplémentaires.	Oui : utilisation de `eksctl` ou d'un modèle de lancement avec une AMI personnalisée	Oui : pour plus d'informations, veuillez consulter les informations d'utilisation du script d'amorçage sur GitHub.	Non
Possibilité d'affecter des adresses IP à des Pods à partir d'un bloc d'adresse CIDR différent de l'adresse IP affectée au nœud.	Oui : utilisation d'un modèle de lancement avec une AMI personnalisée. Pour plus d'informations, consultez Personnalisation des nœuds gérés avec des modèles de lancement.	Oui : pour plus d'informations, consultez Mise en réseau personnalisée pour les pods.	Non
Possibilité d'accéder au nœud via SSH	Oui	Oui	Non – Il n'y a pas de système d'exploitation hôte de nœud auquel se connecter par SSH.
Possibilité de déployer votre propre AMI personnalisée sur les nœuds	Oui : utilisation d'un modèle de lancement	Oui	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Possibilité de déployer votre propre CNI personnalisée sur les nœuds	Oui : utilisation d'un modèle de lancement avec une AMI personnalisée	Oui	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Vous devez mettre à jour l'AMI du nœud par vous-même	Oui : si vous avez déployé une AMI optimisée pour Amazon EKS, vous êtes averti dans la console Amazon EKS lorsque les mises à jour sont disponibles. Vous pouvez effectuer la mise à jour en un clic dans la console. Si vous avez déployé une AMI personnalisée, vous n'êtes pas averti dans la console Amazon EKS lorsque des mises à jour sont disponibles. Vous devez effectuer la mise à jour par vous-même.	Oui, en utilisant des outils autres que la console Amazon EKS. En effet, les nœuds autogérés ne peuvent pas être gérés avec la console Amazon EKS.	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Vous devez mettre à jour la version du nœud Kubernetes par vous-même	Oui : si vous avez déployé une AMI optimisée pour Amazon EKS, vous êtes averti dans la console Amazon EKS lorsque les mises à jour sont disponibles. Vous pouvez effectuer la mise à jour en un clic dans la console. Si vous avez déployé une AMI personnalisée, vous n'êtes pas averti dans la console Amazon EKS lorsque des mises à jour sont disponibles. Vous devez effectuer la mise à jour par vous-même.	Oui, en utilisant des outils autres que la console Amazon EKS. En effet, les nœuds autogérés ne peuvent pas être gérés avec la console Amazon EKS.	Non : vous ne gérez pas les nœuds.
Possibilité d'utiliser le stockage Amazon EBS avec des Pods	Oui	Oui	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Possibilité d'utiliser le stockage Amazon EFS avec des Pods	Oui	Oui	Oui
Possibilité d'utiliser le stockage Amazon FSx pour Lustre avec des Pods	Oui	Oui	Non
Possibilité d'utiliser le Network Load Balancer pour les services	Oui	Oui	Oui, lors de l'utilisation de Créer un équilibreur de charge de réseau
Les pods peuvent s'exécuter dans un sous-réseau public	Oui	Oui	Non
Possibilité d'affecter différents groupes de sécurité VPC à des Pods individuels	Oui : nœuds Linux uniquement	Oui : nœuds Linux uniquement	Oui
Peut exécuter DaemonSets Kubernetes	Oui	Oui	Non
Prend en charge `HostPort` et `HostNetwork` dans le manifeste du Pod	Oui	Oui	Non
Région AWS disponibilité	Toutes les régions prises en charge par Amazon EKS	Toutes les régions prises en charge par Amazon EKS	Quelques régions prises en charge par Amazon EKS
Possibilité d'exécuter des conteneurs sur des hôtes dédiés Amazon EC2	Oui	Oui	Non

Critères	Groupes de nœuds gérés EKS	Nœuds autogérés	AWS Fargate
Tarification	Coût de l'instance Amazon EC2 qui exécute plusieurs Pods. Pour plus d'informations, consultez Tarification Amazon EC2.	Coût de l'instance Amazon EC2 qui exécute plusieurs Pods. Pour plus d'informations, consultez Tarification Amazon EC2.	Coût d'une mémoire Fargate individuelle et d'une configuration CPU. Chaque Pod a son propre coût. Pour en savoir plus, consultez AWS Fargate Tarification.

Groupes de nœuds gérés

Les groupes de nœuds gérés Amazon EKS automatisent l'approvisionnement et la gestion du cycle de vie des nœuds (instances Amazon EC2) pour les clusters Amazon EKS Kubernetes.

Avec les groupes de nœuds gérés Amazon EKS, vous n'avez pas besoin d'approvisionner ou d'enregistrer séparément les instances Amazon EC2 qui fournissent la capacité de calcul pour exécuter vos applications Kubernetes. Vous pouvez créer, mettre à jour ou résilier les nœuds pour votre cluster en une seule opération. Les mises à jour et les résiliations de nœuds purgent automatiquement les nœuds afin de garantir la disponibilité de vos applications.

Chaque nœud géré est approvisionné dans le cadre d'un groupe Amazon EC2 Auto Scaling qui est géré pour vous par Amazon EKS. Chaque ressource, y compris les instances et les groupes Auto Scaling, sont gérées par votre compte AWS . Chaque groupe de nœuds s'exécute dans plusieurs zones de disponibilité que vous définissez.

Vous pouvez ajouter un groupe de nœuds gérés à des clusters nouveaux ou existants à l'aide de la console Amazon EKS`eksctl`, AWS CLI de AWS l'API ou de l'infrastructure sous forme d'outils de code, notamment AWS CloudFormation. Les nœuds lancés dans le cadre d'un groupe de nœuds gérés sont automatiquement marqués pour la découverte automatique par le Kubernetes Cluster

Autoscaler. Vous pouvez utiliser le groupe de nœuds pour appliquer des labels Kubernetes aux nœuds et les mettre à jour à tout moment.

L'utilisation de groupes de nœuds gérés Amazon EKS n'entraîne aucun coût supplémentaire. Vous ne payez que les ressources AWS que vous approvisionnez. Il s'agit notamment des instances Amazon EC2, des volumes Amazon EBS, des heures de cluster Amazon EKS et de toute autre infrastructure. AWS Aucun frais minimum ni aucun engagement initial ne s'appliquent.

Pour démarrer avec un nouveau cluster Amazon EKS et un groupe de nœuds gérés, consultez Commencer à utiliser Amazon EKS — AWS Management Console et AWS CLI.

Pour ajouter un groupe de nœuds gérés à un cluster existant, consultez Création d'un groupe de nœuds gérés.

Concepts des groupes de nœuds gérés

- Les groupes de nœuds gérés Amazon EKS créent et gèrent des instances Amazon EC2 automatiquement.
- Chaque nœud géré est approvisionné dans le cadre d'un groupe Amazon EC2 Auto Scaling qui est géré pour vous par Amazon EKS. De plus, toutes les ressources, y compris les instances Amazon EC2 et les groupes Auto Scaling, sont exécutées au sein de votre AWS compte.
- Le groupe Auto Scaling d'un groupe de nœuds gérés couvre chaque sous-réseau que vous spécifiez lorsque vous créez le groupe.
- Amazon EKS balise les ressources d'un groupe de nœuds gérés de manière à ce qu'elles soient configurées pour utiliser le Cluster Autoscaler de Kubernetes.

> **Important**
>
> Si vous exécutez sur plusieurs zones de disponibilité une application avec état qui est basée sur des volumes Amazon EBS et qui utilise Kubernetes Autoscaling, vous devez configurer plusieurs groupes de nœuds, chacun étant limité à une seule zone de disponibilité. En outre, vous devez activer la fonction `--balance-similar-node-groups`.

- Vous pouvez utiliser un modèle de lancement personnalisé pour un plus grand niveau de flexibilité et de personnalisation lors du déploiement de nœuds gérés. Par exemple, vous pouvez spécifier des arguments `kubelet` supplémentaires et utiliser une AMI personnalisée. Pour plus d'informations, consultez Personnalisation des nœuds gérés avec des modèles de lancement.

Si vous n'utilisez pas de modèle de lancement personnalisé lors de la première création d'un groupe de nœuds gérés, un modèle de lancement est généré automatiquement. Ne modifiez pas manuellement ce modèle généré automatiquement, sinon des erreurs se produiront.

- Amazon EKS suit le modèle de responsabilité partagée pour les CVE et les correctifs de sécurité sur les groupes de nœuds gérés. Lorsque les nœuds gérés exécutent une AMI optimisée pour Amazon EKS, cette dernière est chargée de créer des versions corrigées de l'AMI lorsque des bogues ou des problèmes sont signalés. Nous pouvons publier un correctif. Cependant, vous êtes responsable du déploiement de ces versions corrigées de l'AMI sur vos groupes de nœuds gérés. Lorsque les nœuds gérés exécutent une AMI personnalisée, vous êtes responsable de la création de versions corrigées de l'AMI lorsque des bogues ou des problèmes sont signalés, puis du déploiement de l'AMI. Pour plus d'informations, consultez Mise à jour d'un groupe de nœuds gérés.
- Les groupes de nœuds gérés par Amazon EKS peuvent être lancés dans des sous-réseaux publics et privés. Si vous lancez un groupe de nœuds gérés dans un sous-réseau public à partir du 22 avril 2020, `MapPublicIpOnLaunch` du sous-réseau doit avoir la valeur true pour que les instances puissent rejoindre un cluster. Si le sous-réseau public a été créé à l'aide `eksctl` des AWS CloudFormation modèles Amazon EKS vendus le 26 mars 2020 ou après cette date, ce paramètre est déjà défini sur true. Si les sous-réseaux publics ont été créés avant le 26 mars 2020, vous devez modifier le paramètre manuellement. Pour plus d'informations, consultez Modification de l'attribut d'adressage `IPv4` public de votre sous-réseau.
- Lorsque vous déployez un groupe de nœuds gérés dans des sous-réseaux privés, vous devez vous assurer qu'il peut accéder à Amazon ECR pour extraire des images de conteneurs. Vous pouvez procéder en connectant une passerelle NAT à la table de routage du sous-réseau ou en ajoutant les points de terminaison d'un VPC AWS PrivateLink suivants :
  - Interface de point de terminaison de l'API Amazon ECR : `com.amazonaws.`*`region-code`*`.ecr.api`
  - Interface de point de terminaison de l'API de registre Docker Amazon ECR : `com.amazonaws.`*`region-code`*`.ecr.dkr`
  - Point de terminaison de la passerelle Amazon S3 : `com.amazonaws.`*`region-code`*`.s3`

  Pour d'autres services et points de terminaison couramment utilisés, veuillez consulter la rubrique Exigences relatives aux clusters privés.
- Les groupes de nœuds gérés ne peuvent pas être déployés sur AWS Outposts AWS Wavelength ou dans les Zones AWS Locales.

- Vous pouvez créer plusieurs groupes de nœuds gérés au sein d'un même cluster. Par exemple, vous pouvez créer un groupe de nœuds avec l'AMI Amazon Linux standard optimisée pour Amazon EKS pour certaines charges de travail et un autre avec la variante GPU pour les charges de travail nécessitant un support GPU.
- Si votre groupe de nœuds gérés rencontre un échec de vérification de l'état des instances Amazon EC2, Amazon EKS renvoie un code d'erreur pour vous aider à diagnostiquer le problème. Pour plus d'informations, consultez Codes d'erreurs liées aux groupes de nœuds gérés.
- Amazon EKS ajoute des labels Kubernetes aux instances de groupes de nœuds gérés. Ces labels fournis par Amazon EKS ont le préfixe `eks.amazonaws.com`.
- Amazon EKS purge automatiquement les nœuds à l'aide de l'API Kubernetes lors des résiliations ou des mises à jour.
- Les budgets de perturbation des pods ne sont pas respectés lors de l'arrêt d'un nœud avec `AZRebalance` ou de la réduction du nombre de nœuds souhaité. Ces actions tentent d'expulser Pods dans le nœud. Mais si l'opération prend plus de 15 minutes, le nœud est arrêté, que tous les Pods du nœud aient été arrêtés ou non. Pour prolonger le délai d'arrêt du nœud, ajoutez un hook de cycle de vie au groupe Auto Scaling. Pour plus d'informations, consultez la rubrique Utilisation de hooks de cycle de vie du Guide de l'utilisateur Amazon EC2 Auto Scaling.
- Pour exécuter correctement le processus de vidange après avoir reçu une notification d'interruption ponctuelle ou une notification de rééquilibrage des capacités, `CapacityRebalance` doit être réglé sur `true`.
- La mise à jour des groupes de nœuds gérés respecte les budgets d'interruption Pod que vous avez définis pour vos Pods. Pour plus d'informations, consultez Comportement de mise à jour des nœuds gérés.
- L'utilisation de groupes de nœuds gérés par Amazon EKS est disponible sans coûts supplémentaires. Vous ne payez que pour les AWS ressources que vous fournissez.
- Si vous souhaitez chiffrer les volumes Amazon EBS pour vos nœuds, vous pouvez déployer les nœuds à l'aide d'un modèle de lancement. Pour déployer des nœuds gérés avec des volumes Amazon EBS chiffrés sans utiliser de modèle de lancement, chiffrez tous les nouveaux volumes Amazon EBS créés dans votre compte. Pour plus d'informations, consultez la section Chiffrement par défaut dans le guide de l'utilisateur Amazon EC2.

Types de capacité des groupes de nœuds gérés

Lorsque vous créez un groupe de nœuds gérés, vous pouvez choisir le type de capacité à la demande ou Spot. Amazon EKS déploie un groupe de nœuds gérés avec un groupe Amazon

EC2 Auto Scaling qui contient soit uniquement des instances à la demande, soit uniquement des instances Amazon EC2 Spot. Vous pouvez programmer des Pods pour des applications tolérantes aux pannes dans des groupes de nœuds gérés Spot, et des applications intolérantes aux pannes dans des groupes de nœuds à la demande au sein d'un même cluster Kubernetes. Par défaut, un groupe de nœuds gérés déploie des instances Amazon EC2 à la demande.

À la demande

Avec les instances à la demande, vous payez la capacité de calcul à la seconde, sans engagement à long terme.

Comment ça marche

Par défaut, si vous ne spécifiez pas de Type de capacité, le groupe de nœuds gérés est alloué avec des instances à la demande. Un groupe de nœuds gérés configure un groupe Amazon EC2 Auto Scaling en votre nom avec les paramètres suivants appliqués :

- La stratégie d'allocation pour fournir la capacité à la demande est définie sur `prioritized`. Les groupes de nœuds gérés utilisent l'ordre des types d'instance transmis dans l'API pour déterminer le type d'instance à utiliser en premier lors de la fourniture de la capacité à la demande. Par exemple, vous pouvez spécifier trois types d'instances dans l'ordre suivant : `c5.large`, `c4.large` et `c3.large`. Lorsque vos instances à la demande sont lancées, le groupe de nœuds gérés fournit la capacité à la demande en commençant par `c5.large`, puis `c4.large` et enfin `c3.large`. Pour plus d'informations, consultez Groupe Amazon EC2 Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
- Amazon EKS ajoute le label Kubernetes suivant à tous les nœuds de votre groupe de nœuds gérés qui spécifie le type de capacité :`eks.amazonaws.com/capacityType: ON_DEMAND`. Vous pouvez utiliser ce label pour programmer des applications à état ou intolérantes aux pannes sur des nœuds à la demande.

Spot

Les instances Amazon EC2 Spot sont une capacité Amazon EC2 de réserve qui offre des réductions importantes par rapport aux prix des instances à la demande. Les instances Amazon EC2 Spot peuvent être résiliées avec un avis de résiliation de deux minutes lorsque EC2 a besoin de récupérer la capacité. Pour plus d'informations, consultez la section Instances Spot dans le guide de l'utilisateur Amazon EC2. Vous pouvez configurer un groupe de nœuds gérés avec des instances Amazon EC2 Spot pour optimiser les coûts des nœuds de calcul fonctionnant dans votre cluster Amazon EKS.

Comment ça marche

Pour utiliser des instances Spot dans un groupe de nœuds gérés, créez un groupe de nœuds gérés en définissant le type de capacité comme `spot`. Un groupe de nœuds gérés configure un groupe Amazon EC2 Auto Scaling en votre nom en appliquant les bonnes pratiques Spot suivantes :

- Pour garantir que vos nœuds Spot sont alloués dans les groupes de capacité Spot optimaux, la stratégie d'allocation est définie sur l'une des options suivantes :
  - `price-capacity-optimized` (PCO) : lors de la création de nouveaux groupes de nœuds dans un cluster avec Kubernetes version `1.28` ou supérieure, la stratégie d'allocation est définie sur `price-capacity-optimized`. Cependant, la stratégie d'allocation ne sera pas modifiée pour les groupes de nœuds déjà créés avec `capacity-optimized` avant que les groupes de nœuds gérés par Amazon EKS ne commencent à prendre en charge le PCO.
  - `capacity-optimized` (CO) : lors de la création de nouveaux groupes de nœuds dans un cluster avec Kubernetes version `1.27` ou inférieure, la stratégie d'allocation est définie sur `capacity-optimized`.

  Pour augmenter le nombre de groupes de capacité Spot disponibles pour l'allocation de capacité, configurez un groupe de nœuds gérés pour utiliser plusieurs types d'instances.
- Le rééquilibrage de la capacité Amazon EC2 Spot est activé pour qu'Amazon EKS puisse purger et rééquilibrer vos nœuds Spot proprement, afin de réduire les perturbations des applications lorsqu'un nœud Spot présente un risque élevé d'interruption. Pour plus d'informations, consultez Rééquilibrage de la capacité Amazon EC2 Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
  - Lorsqu'un nœud Spot reçoit une recommandation de rééquilibrage, Amazon EKS tente automatiquement de lancer un nouveau nœud Spot de remplacement.
  - Si un avis d'interruption de deux minutes arrive avant que le nœud Spot de remplacement ne soit dans l'état `Ready`, Amazon EKS commence à purger le nœud Spot qui a reçu la recommandation de rééquilibrage. Amazon EKS vide le nœud dans la mesure du possible. Par conséquent, rien ne garantit qu'Amazon EKS attendra que le nœud de remplacement rejoigne le cluster avant de vider le nœud existant.
  - Lorsqu'un nœud Spot de remplacement est démarré et dans l'état `Ready` sur Kubernetes, Amazon EKS cloisonne et purge le nœud Spot qui a reçu la recommandation de rééquilibrage. Le cloisonnement du nœud Spot garantit que le contrôleur de service n'envoie pas de nouvelles demandes à ce nœud Spot. Il le supprime également de sa liste de nœuds Spot sains et actifs. La purge du nœud Spot garantit que les Pods en cours d'exécution sont expulsés proprement.

- Amazon EKS ajoute le label Kubernetes suivant à tous les nœuds de votre groupe de nœuds gérés qui spécifie le type de capacité :`eks.amazonaws.com/capacityType: SPOT`. Vous pouvez utiliser ce label pour programmer des applications tolérantes aux pannes sur les nœuds Spot.

Considérations relatives à la sélection d'un type de capacité

Lorsque vous décidez de déployer un groupe de nœuds avec une capacité à la demande ou Spot, vous devez tenir compte des conditions suivantes :

- Les instances Spot conviennent bien aux applications sans état, tolérantes aux pannes et flexibles. Il s'agit notamment des charges de travail d'entraînement par lots et de machine learning, des ETL de big data tels qu'Apache Spark, des applications de traitement des files d'attente et des points de terminaison d'API sans état. Étant donné que Spot est une capacité Amazon EC2 de réserve, qui peut changer au fil du temps, nous vous recommandons d'utiliser la capacité Spot pour les charges de travail tolérantes aux interruptions. Plus précisément, la capacité Spot convient aux charges de travail qui peuvent tolérer des périodes où la capacité requise n'est pas disponible.
- Nous vous recommandons d'utiliser la capacité à la demande pour les applications qui sont intolérantes aux pannes. Cela inclut les outils de gestion de cluster tels que les outils de surveillance et d'exploitation, les déploiements qui nécessitent `StatefulSets`, et les applications avec état, telles que les bases de données.
- Pour maximiser la disponibilité de vos applications en utilisant les instances Spot, nous vous recommandons de configurer un groupe de nœuds gérés Spot pour utiliser plusieurs types d'instances. Nous vous recommandons d'appliquer les règles suivantes lorsque vous utilisez plusieurs types d'instance :
  - Dans un groupe de nœuds gérés, si vous utilisez Cluster Autoscaler, nous vous recommandons d'utiliser un ensemble flexible de types d'instances avec la même quantité de ressources vCPU et de mémoire. Ceci afin de garantir que les nœuds de votre cluster soient mis à l'échelle comme prévu. Par exemple, si vous avez besoin de quatre vCPU et de huit GiB de mémoire, utilisez des instances `c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge` ou d'autres types d'instances similaires.
  - Pour améliorer la disponibilité des applications, nous recommandons de déployer plusieurs groupes de nœuds gérés Spot. Pour cela, chaque groupe doit utiliser un ensemble flexible de types d'instances qui disposent des mêmes ressources vCPU et mémoire. Par exemple, si vous avez besoin de 4 vCPU et de 8 GiB de mémoire, nous vous recommandons de créer un groupe de nœuds gérés avec des instances `c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge` ou d'autres types d'instance similaires, et un deuxième groupe

de nœuds gérés avec des instances `m3.xlarge`, `m4.xlarge`, `m5.xlarge`, `m5d.xlarge`, `m5a.xlarge`, `m5n.xlarge` ou d'autres types d'instances similaires.

- Lorsque vous déployez votre groupe de nœuds avec le type de capacité Spot qui utilise un modèle de lancement personnalisé, utilisez l'API pour transmettre plusieurs types d'instances. Ne transmettez pas un seul type d'instance via le modèle de lancement. Pour plus d'informations sur le déploiement d'un groupe de nœuds à l'aide d'un modèle de lancement, consultez Personnalisation des nœuds gérés avec des modèles de lancement.

Création d'un groupe de nœuds gérés

Cette rubrique explique comment lancer des groupes de nœuds gérés Amazon EKS composés de nœuds qui s'enregistrent dans votre cluster Amazon EKS. Une fois que les nœuds ont rejoint le cluster, vous pouvez y déployer des applications Kubernetes.

Si vous lancez un groupe de nœuds gérés par Amazon EKS pour la première fois, nous vous recommandons de suivre l'un de nos guides Démarrer avec Amazon EKS. Les guides fournissent les instructions de création d'un cluster Amazon EKS avec des nœuds.

Important

- Les nœuds Amazon EKS sont des instances Amazon EC2 standard. Vous êtes facturé en fonction des prix Amazon EC2 habituels. Pour plus d'informations, consultez Tarification Amazon EC2.
- Vous ne pouvez pas créer de nœuds gérés dans un Région AWS endroit où vous avez AWS Outposts activé AWS Wavelength les Zones AWS Locales. Vous pouvez créer des nœuds autogérés Région AWS là où vous avez activé dans lequel vous avez AWS Outposts activé AWS Wavelengthles Zones AWS Locales. Pour plus d'informations, consultez Lancement de nœuds Amazon Linux autogérés, Lancement de nœuds Windows autogérés et Lancement de nœuds Bottlerocket autogérés. Vous pouvez également créer un groupe de nœuds Amazon Linux autogéré sur un Outpost. Pour plus d'informations, consultez Lancement de nœuds Amazon Linux autogérés sur un Outpost.
- Si vous ne spécifiez pas un ID d'AMI pour le fichier `bootstrap.sh` inclus dans Linux ou Bottlerocket optimisé pour Amazon EKS, les groupes de nœuds gérés imposent un nombre maximal à la valeur de `maxPods`. Pour les instances avec moins de 30 vCPUs, le nombre maximal est `110`. Pour les instances avec plus de 30 vCPUs, le nombre maximal passe à `250`. Ces chiffres sont basés sur les seuils de capacité de mise à l'échelle de

Kubernetes et les paramètres recommandés lors des tests internes de l'équipe pour la capacité de mise à l'échelle d'Amazon EKS. Pour plus d'informations, consultez l'article de blog Plugin CNI Amazon VPC augmente les limites du nombre de pods par nœud (français non garanti).

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Création d'un cluster Amazon EKS.
- Un rôle IAM existant pour les nœuds à utiliser. Pour en créer un, consultez Rôle IAM de nœud Amazon EKS. Si ce rôle ne comporte aucune des politiques pour le VPC CNI, le rôle distinct suivant est nécessaire pour les pods VPC CNI.
- (Facultatif, mais recommandé) Le module complémentaire Amazon VPC CNI plugin for Kubernetes configuré avec son propre rôle IAM auquel est attachée la politique IAM nécessaire. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
- Être familiarisé avec les considérations énumérées dans Choix d'un type d'instance Amazon EC2. Selon le type d'instance que vous choisissez, il peut y avoir des prérequis supplémentaires pour votre cluster et votre VPC.
- Pour ajouter un groupe de nœuds gérés Windows, vous devez d'abord activer la prise en charge de Windows pour votre cluster. Pour plus d'informations, consultez Activation de la prise en charge de Windows pour votre cluster Amazon EKS.

Vous pouvez créer un groupe de nœuds gérés avec `eksctl` ou la AWS Management Console.

eksctl

Pour créer un groupe de nœuds gérés **`eksctl`**

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

1. (Facultatif) Si la politique d'IAM gérée AmazonEKS_CNI_Policy est associée à votre Rôle IAM de nœud Amazon EKS, nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service Kubernetes `aws-node` à la place. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
2. Créez un groupe de nœuds gérés avec ou sans modèle de lancement personnalisé. La spécification manuelle d'un modèle de lancement permet une plus grande personnalisation d'un groupe de nœuds. Par exemple, cela peut permettre de déployer une AMI personnalisée ou de fournir des arguments au script `boostrap.sh` dans une AMI optimisée par Amazon EKS. Pour obtenir la liste complète de toutes les options disponibles et des valeurs par défaut, saisissez la commande suivante.

```
eksctl create nodegroup --help
```

Dans la commande suivante, remplacez *`my-cluster`* par le nom de votre cluster et remplacez *`my-mng`* par le nom de votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.

> **Important**
>
> Si vous n'utilisez pas de modèle de lancement personnalisé lors de la première création d'un groupe de nœuds gérés, n'en utilisez pas un ultérieurement pour ce groupe de nœuds. Si vous n'avez pas spécifié de modèle de lancement personnalisé, le système génère automatiquement un modèle de lancement que nous ne vous recommandons pas de modifier manuellement. La modification manuelle de ce modèle de lancement généré automatiquement peut entraîner des erreurs.

Sans modèle de lancement

`eksctl` crée un modèle de lancement Amazon EC2 par défaut dans votre compte et déploie le groupe de nœuds à l'aide d'un modèle de lancement qu'il crée en fonction des options que vous spécifiez. Avant de spécifier une valeur pour `--node-type`, consultez Choix d'un type d'instance Amazon EC2.

Remplacez *ami-family* par un mot clé autorisé. Pour plus d'informations, consultez Définition de la famille AMI de nœuds dans la documentation `eksctl`. Remplacez *my-key* par le nom de votre paire de clés Amazon EC2 ou de votre clé publique. Cette clé est utilisée pour SSH dans vos nœuds après leur lancement.

> **Note**
>
> Sous Windows, cette commande n'active pas SSH. Elle associe votre key pair Amazon EC2 à l'instance et vous permet d'accéder à l'instance.

Si vous ne possédez pas déjà une paire de clés Amazon EC2, vous pouvez en créer une dans l' AWS Management Console. Pour Linux plus d'informations, consultez les paires de clés et Linux instances Amazon EC2 dans le guide de l'utilisateur Amazon EC2. Pour Windows plus d'informations, consultez les paires de clés et Windows instances Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

- Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
- Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Si vous souhaitez bloquer l'accès des Pod à IMDS, ajoutez l'option **`--disable-pod-imds`** à la commande suivante.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --region region-code \
  --name my-mng \
  --node-ami-family ami-family \
  --node-type m5.large \
  --nodes 3 \
```

```
--nodes-min 2 \
--nodes-max 4 \
--ssh-access \
--ssh-public-key my-key
```

Vos instances peuvent éventuellement attribuer un nombre significativement plus élevé d'adresses IP aux Pods, attribuer des adresses IP aux Pods à partir d'un bloc CIDR différent de celui de l'instance, et être déployées dans un cluster sans accès Internet. Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2, Mise en réseau personnalisée pour les pods et Exigences relatives aux clusters privés pour connaître les options supplémentaires à ajouter à la commande précédente.

Les groupes de nœuds gérés calculent et appliquent une valeur unique pour le nombre maximum de Pods pouvant s'exécuter sur chaque nœud de votre groupe de nœuds, en fonction du type d'instance. Si vous créez un groupe de nœuds avec différents types d'instances, la plus petite valeur calculée sur tous les types d'instances est appliquée comme nombre maximal de Pods pouvant s'exécuter sur chaque type d'instance du groupe de nœuds. Les groupes de nœuds gérés calculent la valeur à l'aide du script référencé dans Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2.

Avec modèle de lancement

Le modèle de lancement doit déjà exister et répondre aux exigences spécifiées dans Concepts de base de la configuration d'un modèle de lancement.

Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

- Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
- Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Si vous souhaitez bloquer l'accès des Pod à l'IMDS, spécifiez les paramètres nécessaires dans le modèle de lancement.

a. Copiez les contenus suivants sur votre appareil. Remplacez les *example values* puis exécutez la commande modifiée pour créer le fichier `eks-nodegroup.yaml`. Plusieurs paramètres que vous spécifiez lors d'un déploiement sans modèle de lancement sont déplacés dans le modèle de lancement. Si vous ne spécifiez pas de `version`, la version par défaut est utilisée.

```
cat >eks-nodegroup.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
managedNodeGroups:
- name: my-mng
  launchTemplate:
    id: lt-id
    version: "1"
EOF
```

Pour obtenir la liste complète des paramètres du fichier de configuration `eksctl`, consultez Schéma de fichier de configuration dans la documentation `eksctl`. Vos instances peuvent éventuellement attribuer un nombre significativement plus élevé d'adresses IP aux Pods, attribuer des adresses IP aux Pods à partir d'un bloc CIDR différent de celui de l'instance, utiliser l'environnement d'exécution `containerd` et être déployées dans un cluster sans accès Internet sortant. Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2, Mise en réseau personnalisée pour les pods, Testez la migration Docker de `containerd` et Exigences relatives aux clusters privés pour connaître les options supplémentaires à ajouter au fichier de configuration.

Si vous n'avez pas spécifié d'ID d'AMI dans votre modèle de lancement, les groupes de nœuds gérés calculent et appliquent une valeur unique pour le nombre maximal de Pods pouvant s'exécuter sur chaque nœud de votre groupe de nœuds, en fonction du type d'instance. Si vous créez un groupe de nœuds avec différents types d'instances, la plus petite valeur calculée sur tous les types d'instances est appliquée comme nombre maximal de Pods pouvant s'exécuter sur chaque type d'instance du groupe de nœuds.

Les groupes de nœuds gérés calculent la valeur à l'aide du script référencé dans Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2.

Si vous avez spécifié un ID d'AMI dans votre modèle de lancement, spécifiez le nombre maximal de Pods pouvant s'exécuter sur chaque nœud de votre groupe de nœuds si vous utilisez un réseau personnalisé ou si vous voulez augmenter le nombre d'adresses IP attribuées à votre instance. Pour plus d'informations, consultez Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2.

b. Déployez le groupe de nœuds avec la commande suivante.

```
eksctl create nodegroup --config-file eks-nodegroup.yaml
```

AWS Management Console

Pour créer un groupe de nœuds géré à l'aide du AWS Management Console

1. Attendez que le statut de votre cluster s'affiche soit `ACTIVE`. Vous ne pouvez pas créer un groupe de nœuds gérés pour un cluster qui n'est pas déjà `ACTIVE`.
2. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
3. Choisissez le nom du cluster dans lequel vous souhaitez créer un groupe de nœuds gérés.
4. Sélectionner l'onglet Calcul.
5. Choisissez Ajouter un groupe de nœuds.
6. Sur la page Configurer un groupe de nœuds, définissez les paramètres en conséquence, puis choisissez Next (Suivant).

   - Nom : saisissez un nom unique pour votre groupe de nœuds gérés. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.
   - Rôle IAM de nœud : choisissez le rôle d'instance de nœud à utiliser avec votre groupe de nœuds. Pour plus d'informations, consultez Rôle IAM de nœud Amazon EKS.

> **Important**
>
> - Vous ne pouvez pas utiliser le même rôle que celui utilisé pour créer des clusters.
> - Nous vous recommandons d'utiliser un rôle qui n'est pas actuellement utilisé par un groupe de nœuds autogérés. Sinon, vous prévoyez de l'utiliser avec un nouveau groupe de nœuds autogérés. Pour plus d'informations, consultez Suppression d'un groupe de nœuds gérés.

- Utiliser le modèle de lancement : (facultatif) choisissez si vous souhaitez utiliser un modèle de lancement existant. Sélectionnez un Nom de modèle de lancement. Sélectionnez ensuite une Version du modèle de lancement. Si vous ne sélectionnez pas de version, Amazon EKS utilise la version par défaut du modèle. Les modèles de lancement permettent de personnaliser davantage votre groupe de nœuds, par exemple en vous permettant de déployer une AMI personnalisée, d'attribuer un nombre nettement plus élevé d'adresses IP aux Pods, d'attribuer aux Pods des adresses IP provenant d'un bloc CIDR différent de celui de l'instance, d'activer l'environnement d'exécution `containerd` pour vos instances et de déployer des nœuds dans un cluster sans accès Internet sortant. Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2, Mise en réseau personnalisée pour les pods, Testez la migration Docker de `containerd` et Exigences relatives aux clusters privés.

  Le modèle de lancement doit répondre aux exigences décrites dans Personnalisation des nœuds gérés avec des modèles de lancement. Si vous n'utilisez pas votre propre modèle de lancement, l'API Amazon EKS crée un modèle de lancement Amazon EC2 par défaut dans votre compte et déploie le groupe de nœuds à l'aide du modèle de lancement par défaut.

  Si vous mettez en œuvre des rôles IAM pour les comptes de service et attribuez les autorisations nécessaires directement à chaque Pod qui requiert l'accès aux services AWS et qu'aucun Pods de votre cluster ne requiert l'accès à IMDS pour d'autres raisons, comme la récupération de la Région AWS actuelle, vous pouvez également désactiver l'accès à IMDS pour les Pods qui n'utilisent pas de réseau hôte dans un modèle de lancement. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

- Étiquettes Kubernetes : (facultatif) vous pouvez choisir d'appliquer des étiquettes Kubernetes aux nœuds de votre groupe de nœuds gérés.
- Rejets Kubernetes : (facultatif) vous pouvez choisir d'appliquer des rejets Kubernetes aux nœuds de votre groupe de nœuds gérés. Les options disponibles dans le menu Effet sont **`NoSchedule`**, **`NoExecute`**, et **`PreferNoSchedule`**. Pour plus d'informations, consultez Rejets de nœuds sur les groupes de nœuds gérés.
- Identifications : (facultatif) vous pouvez choisir d'identifier votre groupe de nœuds gérés Amazon EKS. Ces identifications ne se propagent pas aux autres ressources du groupe de nœuds, telles que les groupes ou instances Auto Scaling. Pour plus d'informations, consultez Étiquetage de vos ressources Amazon EKS.

7. Sur la page Définir la configuration de calcul et de mise à l'échelle, définissez les paramètres en conséquence, puis choisissez Next (Suivant).

   - Type d'AMI : sélectionnez un type d'AMI. Si vous déployez des instances Arm, veillez à tenir compte des considérations contenues dans AMI Amazon Linux Arm optimisées pour Amazon EKS avant le déploiement.

     Si vous avez spécifié un modèle de lancement à la page précédente, et spécifié une AMI dans le modèle de lancement, vous ne pouvez sélectionner aucune valeur. La valeur du modèle s'affiche. L'AMI spécifiée dans le modèle doit répondre aux exigences décrites dans Spécification d'une AMI.
   - Type de capacité : sélectionnez un type de capacité. Pour plus d'informations sur le choix d'un type de capacité, consultez Types de capacité des groupes de nœuds gérés. Vous ne pouvez pas combiner différents types de capacités au sein d'un même groupe de nœuds. Si vous souhaitez utiliser les deux types de capacités, créez des groupes de nœuds distincts, chacun avec son propre type de capacité et d'instance.
   - Types d'instance : par défaut, un ou plusieurs types d'instances sont spécifiés. Pour supprimer un type d'instance par défaut, sélectionnez la croix (X) sur le côté droit du type d'instance. Choisissez les types d'instances à utiliser dans votre groupe de nœuds gérés. Pour plus d'informations, consultez Choix d'un type d'instance Amazon EC2.

     La console affiche un ensemble de types d'instances couramment utilisés. Si vous devez créer un groupe de nœuds gérés avec un type d'instance qui n'est pas affiché, utilisez `eksctl`, la AWS CLI, AWS CloudFormation ou un kit SDK pour créer le groupe de nœuds. Si vous avez spécifié un modèle de lancement à la page précédente, vous ne pouvez pas sélectionner une valeur car le type d'instance doit être spécifié dans le modèle de

lancement. La valeur du modèle de lancement s'affiche. Si vous avez sélectionné Spot pour le type de capacité, nous vous recommandons de spécifier plusieurs types d'instances pour améliorer la disponibilité.

- Taille du disque : saisissez la taille du disque (en GiB) à utiliser pour le volume racine du nœud.

  Si vous avez spécifié un modèle de lancement à la page précédente, vous ne pouvez pas sélectionner une valeur, car elle doit être spécifiée dans le modèle de lancement.

- Taille souhaitée : spécifiez le nombre actuel de nœuds que le groupe de nœuds gérés doit conserver au lancement.

  > **Note**
  >
  > Amazon EKS n'augmente pas ou ne réduit pas automatiquement votre groupe de nœuds. Cependant, vous pouvez configurer le Cluster Autoscaler Kubernetes pour qu'il le fasse pour vous.

- Taille minimale : spécifiez le nombre minimal de nœuds vers lequel le groupe de nœuds gérés peut être mis à l'échelle.
- Taille maximale : spécifiez le nombre maximal de nœuds vers lequel le groupe de nœuds gérés peut être mis à niveau.
- Configuration des mises à jour du groupe : (facultatif) vous pouvez sélectionner le nombre ou le pourcentage de nœuds à mettre à jour en parallèle. Ces nœuds seront indisponibles pendant la mise à jour. Pour Maximum non disponible, sélectionnez l'une des options suivantes et spécifiez une valeur :
  - Nombre : sélectionnez et spécifiez le nombre de nœuds de votre groupe de nœuds pouvant être mis à jour en parallèle.
  - Pourcentage : sélectionnez et spécifiez le pourcentage de nœuds de votre groupe de nœuds pouvant être mis à jour en parallèle. Cela est pratique si votre groupe de nœuds contient de nombreux nœuds.

8. Sur la page Spécifier les détails, définissez les paramètres en conséquence, puis choisissez Next (Suivant).

   - Sous-réseaux : choisissez les sous-réseaux dans lesquels vous souhaitez lancer vos nœuds gérés.

> **Important**
>
> Si vous exécutez sur plusieurs zones de disponibilité une application avec état qui est basée sur des volumes Amazon EBS et qui utilise Kubernetes Autoscaling, vous devez configurer plusieurs groupes de nœuds, chacun étant limité à une seule zone de disponibilité. En outre, vous devez activer la fonction `--balance-similar-node-groups`.

> **Important**
>
> - Si vous choisissez un sous-réseau public et que seul le point de terminaison du serveur d'API publique est activé, le paramètre `MapPublicIPOnLaunch` du sous-réseau doit avoir la valeur `true` pour que les instances rejoignent un cluster. Si le sous-réseau a été créé à l'aide de `eksctl` ou des modèles AWS CloudFormation vendus pa Amazon EKS ou à partir du 26 mars 2020, ce paramètre a déjà la valeur `true`. Si les sous-réseaux ont été créés avec les AWS CloudFormation modèles `eksctl` ou avant le 26 mars 2020, vous devez modifier le paramètre manuellement. Pour plus d'informations, consultez Modification de l'attribut d'adressage `IPv4` public de votre sous-réseau.
> - Si vous utilisez un modèle de lancement et spécifiez plusieurs interfaces réseau, Amazon EC2 n'attribue pas automatiquement une adresse `IPv4` publique, même si `MapPublicIpOnLaunch` a la valeur `true`. Pour que les nœuds puissent rejoindre le cluster dans ce scénario, vous devez soit activer le point de terminaison du serveur d'API privée du cluster, soit lancer les nœuds dans un sous-réseau privé avec un accès Internet sortant fourni par une méthode alternative, telle qu'une passerelle NAT. Pour plus d'informations, consultez la section Adressage IP des instances Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

- Configurez l'accès SSH aux nœuds (facultatif). L'activation de SSH vous permet de vous connecter à vos instances et de recueillir des informations de diagnostic en cas de problème. Nous vous recommandons vivement d'activer l'accès à distance lorsque vous créez un groupe de nœuds. Vous ne pouvez pas activer l'accès distant une fois le groupe de nœuds créé.

Si vous avez choisi d'utiliser un modèle de lancement, cette option n'est pas affichée. Pour autoriser l'accès à distance pour vos noeuds, spécifiez une paire de clés dans le modèle de lancement et assurez-vous que le port approprié est ouvert aux nœuds des groupes de sécurité que vous spécifiez dans le modèle de lancement. Pour plus d'informations, consultez Utilisation des groupes de sécurité.

> **Note**
>
> Sous Windows, cette commande n'active pas SSH. Elle associe votre key pair Amazon EC2 à l'instance et vous permet d'accéder à l'instance.

- Pour Paire de clés SSH (facultatif), choisissez une clé SSH Amazon EC2 à utiliser. Pour Linux plus d'informations, consultez les paires de clés et Linux instances Amazon EC2 dans le guide de l'utilisateur Amazon EC2. Pour Windows plus d'informations, consultez les paires de clés et Windows instances Amazon EC2 dans le guide de l'utilisateur Amazon EC2. Si vous avez choisi d'utiliser un modèle de lancement, vous ne pouvez pas en sélectionner un. Lorsqu'une clé SSH Amazon EC2 est fournie pour les groupes de nœuds utilisant les AMI Bottlerocket, le conteneur d'administration est également activé. Pour plus d'informations, consultez Conteneur d'administration sur GitHub.
- Pour Autoriser l'accès à distance SSH depuis, si vous souhaitez limiter l'accès à des instances spécifiques, sélectionnez les groupes de sécurité associés à ces instances. Si vous ne sélectionnez pas de groupes de sécurité spécifiques, l'accès SSH est autorisé à partir de n'importe où sur Internet (`0.0.0.0/0`).

9. Sur la page Vérifier et créer, vérifiez la configuration de votre groupe de nœuds gérés et choisissez Create (Créer).

   Si les nœuds ne parviennent pas à rejoindre le cluster, reportez-vous à Les nœuds ne parviennent pas à joindre le cluster dans le guide de dépannage.

10. Observez le statut de vos nœuds et attendez qu'ils obtiennent le statut `Ready`.

    ```
    kubectl get nodes --watch
    ```

11. (Nœuds GPU uniquement) Si vous avez choisi un type d'instance GPU et l'AMI accélérée optimisée pour Amazon EKS, alors vous devez mettre en œuvre le plugin de périphérique NVIDIA pour Kubernetes en tant que DaemonSet dans votre cluster. Remplacez *vX.X.X* par la version NVIDIA/k8s-device-plugin souhaitée avant d'exécuter la commande suivante.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-
plugin/vX.X.X/nvidia-device-plugin.yml
```

Maintenant que vous disposez d'un cluster Amazon EKS fonctionnel avec des nœuds, vous êtes prêt à commencer à installer les modules complémentaires Kubernetes et à déployer des applications sur votre cluster. Les rubriques suivantes de la documentation vous aideront à étendre les fonctionnalités de votre cluster.

- Le principal IAM qui a créé le cluster est le seul principal à pouvoir effectuer des appels au serveur d'API Kubernetes à l'aide de `kubectl` ou de la AWS Management Console. Si vous souhaitez que d'autres principaux IAM aient accès à votre cluster, vous devez les ajouter. Pour plus d'informations, consultez Autoriser l'accès aux Kubernetes API et Autorisations nécessaires.
- Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :
  - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
  - Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

  Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.
- Autoscaling : configurez l'outil Cluster Autoscaler de Kubernetes pour ajuster automatiquement le nombre de nœuds dans vos groupes de nœuds.
- Déployez un exemple d'application sur votre cluster.
- Gestion du cluster : apprenez à utiliser des outils importants pour gérer votre cluster.

Mise à jour d'un groupe de nœuds gérés

Lorsque vous lancez une mise à jour du groupe de nœuds gérés, Amazon EKS met à jour vos nœuds automatiquement en effectuant les étapes énumérées dans Comportement de mise à jour des nœuds gérés. Si vous utilisez une AMI optimisée pour Amazon EKS, Amazon EKS applique automatiquement les derniers correctifs de sécurité et les mises à jour du système d'exploitation à vos nœuds dans le cadre de la dernière version de l'AMI.

Il existe plusieurs scénarios dans lesquels il est utile de mettre à jour la version ou la configuration de votre groupe de nœuds gérés Amazon EKS :

- Vous avez mis à jour la version de Kubernetes de votre cluster Amazon EKS et vous souhaitez mettre à jour vos nœuds pour utiliser la même version de Kubernetes.
- Une nouvelle version d'AMI est disponible pour votre groupe de nœuds gérés. Pour plus d'informations sur les versions d'AMI, consultez ces sections :
  - Versions d'AMI Amazon Linux optimisées pour Amazon EKS
  - AMI Bottlerocket optimisées pour Amazon EKS
  - Versions d'AMI Windows optimisée pour Amazon EKS
- Vous souhaitez ajuster le nombre minimum, maximum ou souhaité d'instances dans votre groupe de nœuds gérés.
- Vous voulez ajouter ou supprimer les labels Kubernetes des instances de votre groupe de nœuds gérés.
- Vous souhaitez ajouter ou supprimer des AWS balises dans votre groupe de nœuds gérés.
- Vous devez déployer une nouvelle version d'un modèle de lancement avec des modifications de configuration, telles qu'une AMI personnalisée mise à jour.
- Vous avez déployé une version `1.9.0` ou une version ultérieure du module complémentaire Amazon VPC CNI, activé le module complémentaire pour la délégation de préfixes et souhaitez que les nouvelles AWS Nitro System instances d'un groupe de nœuds prennent en charge un nombre considérablement accru de. Pods Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.
- Vous avez activé la délégation de préfixes IP pour les nœuds Windows et souhaitez que les nouvelles instances AWS Nitro System d'un groupe de nœuds prennent en charge un nombre considérablement accru de. Pods Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.

S'il existe une version plus récente de l'AMI pour la version Kubernetes de votre groupe de nœuds gérés, vous pouvez mettre à jour la version de votre groupe de nœuds pour utiliser la nouvelle version de l'AMI. De même, si votre cluster exécute une version de Kubernetes plus récente que votre groupe de nœuds, vous pouvez mettre à jour le groupe de nœuds afin d'utiliser la dernière version de l'AMI correspondant à la version de Kubernetes de votre cluster.

Lorsqu'un nœud d'un groupe de nœuds gérés est résilié en raison d'une opération de mise à l'échelle ou d'une mise à jour, les Pods de ce nœud sont purgés en premier. Pour plus d'informations, consultez Comportement de mise à jour des nœuds gérés.

Mise à jour d'une version de groupe de nœuds

Vous pouvez mettre à jour une version de groupe de nœuds avec `eksctl` ou la AWS Management Console. La version que vous mettez à jour ne peut pas être supérieure à celle du plan de contrôle.

eksctl

Pour mettre à jour une version de groupe de nœuds avec **`eksctl`**

- Mettez à jour un groupe de nœuds gérés vers la dernière version AMI de la même version de Kubernetes qui est actuellement déployée sur les nœuds avec la commande suivante. Remplacez chaque *`example value`* par vos propres valeurs.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code
```

> **Note**
>
> Si vous mettez à niveau un groupe de nœuds déployé avec un modèle de lancement vers une nouvelle version du modèle de lancement, ajoutez `--launch-template-version` *`version-number`* à la commande précédente. Le modèle de lancement doit répondre aux exigences décrites dans Personnalisation des nœuds gérés avec des modèles de lancement. Si le modèle de lancement inclut une AMI personnalisée, l'AMI doit répondre aux exigences décrites dans Spécification d'une AMI. Lorsque vous mettez à niveau votre groupe de nœuds vers une version plus récente de votre modèle de lancement, chaque nœud est recyclé pour correspondre à la nouvelle configuration de la version du modèle de lancement qui a été spécifiée.
> Vous ne pouvez pas mettre directement à niveau un groupe de nœuds déployé sans modèle de lancement vers une nouvelle version du modèle de lancement. À la place, vous devez déployer un nouveau groupe de nœuds en utilisant le modèle de lancement pour mettre à jour le groupe de nœuds vers une nouvelle version du modèle de lancement.

Vous pouvez mettre à niveau un groupe de nœuds vers la même version que la version de Kubernetes du plan de contrôle. Par exemple, si vous avez un cluster exécutant Kubernetes

1.29, vous pouvez mettre à niveau les nœuds exécutant actuellement Kubernetes 1.28 vers la version 1.29 à l'aide de la commande suivante.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code \
  --kubernetes-version=1.29
```

AWS Management Console

Pour mettre à jour la version d'un groupe de nœuds avec AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le cluster qui contient le groupe de nœuds à mettre à jour.
3. Si une mise à jour est disponible pour au moins un groupe de nœuds, une boîte de dialogue apparaît en haut de la page pour vous en informer. Si vous sélectionnez l'onglet Compute (Calcul), vous verrez Update now (Mettre à jour maintenant) dans la colonne AMI release version (Version de l'AMI) du tableau Node Groups (Groupes de nœuds) pour le groupe de nœuds pour lequel une mise à jour est disponible. Pour mettre à jour le groupe de nœuds, sélectionnez Update now (Mettre à jour maintenant).

   Vous ne verrez pas de notification pour les groupes de nœuds qui ont été déployés avec une AMI personnalisée. Si vos nœuds sont déployés avec une AMI personnalisée, effectuez les étapes suivantes pour déployer une nouvelle AMI personnalisée mise à jour.

   a. Créez une version de votre AMI.

   b. Créez une version de modèle de lancement avec le nouvel ID d'AMI.

   c. Mettez à niveau les nœuds vers la nouvelle version du modèle de lancement.
4. Dans la boîte de dialogue Update node group version (Mettre à jour la version du groupe de nœuds), activez ou désactivez les options suivantes :

   - Update node group version (Mettre à jour la version du groupe de nœuds) : cette option n'est pas disponible si vous avez déployé une AMI personnalisée ou que votre AMI optimisée pour Amazon EKS correspond actuellement à la dernière version de votre cluster.

- Change launch template version (Modifier la version du modèle de lancement) : cette option n'est pas disponible si le groupe de nœuds est déployé sans modèle de lancement personnalisé. Vous pouvez uniquement mettre à jour la version du modèle de lancement pour un groupe de nœuds qui a été déployé avec un modèle de lancement personnalisé. Sélectionnez la Version du modèle de lancement vers laquelle vous souhaitez mettre à jour le groupe de nœuds. Si votre groupe de nœuds est configuré avec une AMI personnalisée, la version que vous sélectionnez doit également spécifier une AMI. Lorsque vous passez à une version plus récente de votre modèle de lancement, chaque nœud est recyclé pour correspondre à la nouvelle configuration de la version du modèle de lancement spécifiée.

5. Pour Update strategy (Politique de mise à jour), sélectionnez l'une des options suivantes :

   - Mise à jour continue : cette option respecte les budgets d'interruption des Pod de votre cluster. Les mises à jour échouent s'il existe un problème de budget d'interruption de Pod qui empêche Amazon EKS de purger proprement les Pods qui fonctionnent sur ce groupe de nœuds.
   - Forcer la mise à jour : cette option ne respecte pas les budgets d'interruption des Pod. Les mises à jour se produisent indépendamment des problèmes de budget d'interruption des Pod en forçant le redémarrage des nœuds.

6. Choisissez Mettre à jour.

Modification de la configuration d'un groupe de nœuds

Vous pouvez modifier une partie de la configuration d'un groupe de nœuds gérés.

Pour modifier la configuration d'un groupe de nœuds

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le cluster qui contient le groupe de nœuds à modifier.
3. Sélectionnez l'onglet Compute (Calcul).
4. Sélectionnez le groupe de nœuds à modifier, puis choisissez Edit (Modifier).
5. (Facultatif) Sur la page Edit node group (Modifier le groupe de nœuds), effectuez les opérations suivantes :

   a. Modifiez la Node group scaling configuration (Configuration de mise à l'échelle du groupe de nœuds).

- Taille souhaitée : spécifiez le nombre actuel de nœuds que le groupe de nœuds gérés doit conserver au lancement.
- Taille minimale : spécifiez le nombre minimal de nœuds vers lequel le groupe de nœuds gérés peut être mis à l'échelle.
- Taille maximale : spécifiez le nombre maximal de nœuds vers lequel le groupe de nœuds gérés peut être mis à niveau. Pour connaître le nombre maximal de noeuds pris en charge dans un groupe de nœuds, consultez Service quotas Amazon EKS.

b. (Facultatif) Ajoutez ou supprimez des labels Kubernetes aux nœuds de votre groupe de nœuds. Les labels affichés ici sont uniquement ceux que vous avez appliqués avec Amazon EKS. D'autres labels peuvent exister sur vos nœuds qui n'apparaissent pas ici.

c. (Facultatif) Ajoutez ou supprimez des rejets Kubernetes aux nœuds de votre groupe de nœuds. Les taints ajoutés peuvent avoir l'effet de **`NoSchedule`**, **`NoExecute`** ou **`PreferNoSchedule`**. Pour plus d'informations, consultez Rejets de nœuds sur les groupes de nœuds gérés.

d. (Facultatif) Ajoutez ou supprimez des Tags (Balises) de la ressource de votre groupe de nœuds. Ces identifications ne sont appliquées qu'au groupe de nœuds Amazon EKS. Elle ne se propagent pas aux autres ressources, telles que les sous-réseaux ou les instances Amazon EC2 dans le groupe de nœuds.

e. (Facultatif) Modifiez la Configuration des mises à jour du groupe. Sélectionnez Number (Nombre) ou Percentage (Pourcentage).

- Nombre : sélectionnez et spécifiez le nombre de nœuds de votre groupe de nœuds pouvant être mis à jour en parallèle. Ces nœuds ne seront pas disponibles pendant la mise à jour.
- Pourcentage : sélectionnez et spécifiez le pourcentage de nœuds de votre groupe de nœuds pouvant être mis à jour en parallèle. Ces nœuds ne seront pas disponibles pendant la mise à jour. Ceci est utile si vous avez beaucoup de nœuds dans votre groupe de nœuds.

f. Lorsque vous avez terminé les modifications, choisissez Save changes (Enregistrer les modifications).

Comportement de mise à jour des nœuds gérés

La stratégie de mise à jour des composants master d'Amazon EKS comporte quatre phases différentes décrites dans les sections suivantes.

Phase de configuration

La phase de configuration comporte les étapes suivantes :

1. Une nouvelle version du modèle de lancement Amazon EC2 est créée pour le groupe Auto Scaling associé à votre groupe de nœuds. La nouvelle version du modèle de lancement utilise l'AMI cible ou une version personnalisée du modèle de lancement pour la mise à jour.
2. Le groupe Auto Scaling est mis à jour pour utiliser la dernière version du modèle de lancement.
3. La quantité maximale de nœuds à mettre à niveau en parallèle est déterminée à l'aide de la propriété `updateConfig` pour le groupe de nœuds. Le maximum indisponible a un quota de 100 nœuds. La valeur par défaut est de un nœud. Pour plus d'informations, consultez la propriété `updateConfig` dans la référence d'API Amazon EKS.

Phase d'augmentation d'échelle

Lors de la mise à niveau des nœuds d'un groupe de nœuds gérés, les nœuds mis à niveau sont lancés dans la même zone de disponibilité que ceux qui sont mis à niveau. Pour garantir ce placement, nous utilisons le rééquilibrage de la zone de disponibilité d'Amazon EC2. Pour plus d'informations, consultez Rééquilibrage de la zone de disponibilité dans le Guide de l'utilisateur Amazon EC2 Auto Scaling. Pour répondre à cette exigence, il est possible que nous lancions jusqu'à deux instances par zone de disponibilité dans votre groupe de nœuds gérés.

La phase d'augmentation d'échelle comporte les étapes suivantes :

1. La taille maximale et la taille souhaitée du groupe Auto Scaling sont augmentées en fonction de la plus grande des deux valeurs suivantes :
   - Jusqu'à deux fois le nombre de zones de disponibilité dans lesquelles le groupe Auto Scaling est déployé.
   - Le maximum indisponible de la mise à niveau.

     Par exemple, si votre groupe de nœuds a cinq zones de disponibilité et que `maxUnavailable` est égal à un, le processus de mise à niveau peut lancer un maximum de 10 nœuds.

Cependant, lorsqu'il `maxUnavailable` est égal à 20 (ou à une valeur supérieure à 10), le processus lancera 20 nouveaux nœuds.

2. Après la mise à l'échelle du groupe Auto Scaling, le processus vérifie si les nœuds utilisant la dernière configuration sont présents dans le groupe de nœuds. Cette étape ne réussit que si elle répond à ces critères :
   - Au moins un nouveau nœud est lancé dans chaque zone de disponibilité où le nœud existe.
   - Chaque nouveau nœud doit être dans l'état `Ready`.
   - Les nouveaux nœuds doivent avoir des étiquettes Amazon EKS appliquées.

     Il s'agit des étiquettes Amazon EKS appliquées sur les composants master d'un groupe de nœuds réguliers :

     - `eks.amazonaws.com/nodegroup-image=`*`$amiName`*
     - `eks.amazonaws.com/nodegroup=`*`$nodeGroupName`*

     Il s'agit des étiquettes appliquées par Amazon EKS sur les composants master dans un modèle de lancement personnalisé ou un groupe de composants AMI :

     - `eks.amazonaws.com/nodegroup-image=`*`$amiName`*
     - `eks.amazonaws.com/nodegroup=`*`$nodeGroupName`*
     - `eks.amazonaws.com/sourceLaunchTemplateId=`*`$launchTemplateId`*
     - `eks.amazonaws.com/sourceLaunchTemplateVersion=`*`$launchTemplateVersion`*
3. Les nœuds sont marqués non planifiables pour éviter la planification de nouveaux Pods. Les nœuds `node.kubernetes.io/exclude-from-external-load-balancers=true` sont également étiquetés pour être supprimés des équilibreurs de charge avant d'être arrêtés.

Les raisons suivantes sont connues pour provoquer une erreur `NodeCreationFailure` dans cette phase :

Capacité insuffisante dans la zone de disponibilité

Il est possible que la zone de disponibilité n'ait pas la capacité des types d'instance demandés. Il est recommandé de configurer plusieurs types d'instances lors de la création d'un groupe de nœuds gérés.

Limites d'instance EC2 sur votre compte

Vous pouvez être amené à augmenter le nombre d'instances Amazon EC2 que votre compte peut exécuter simultanément en utilisant Service Quotas. Pour plus d'informations, consultez EC2

Service Quotas dans le Guide de l'utilisateur Amazon Elastic Compute Cloud pour les instances Linux.

Données utilisateur personnalisées

Les données utilisateur personnalisées peuvent parfois interrompre le processus d'amorçage. Ce scénario peut conduire à ce que le `kubelet` ne démarre pas sur le nœud ou que les nœuds n'obtiennent pas les étiquettes Amazon EKS attendues. Pour plus d'informations, consultez Spécification d'une AMI.

Toute modification qui rend un nœud défectueux ou pas prêt

La pression du disque sur le nœud, la pression de la mémoire et des conditions similaires peuvent empêcher un nœud de passer à l'état `Ready`.

Phase de mise à niveau

La phase de mise à niveau comporte les étapes suivantes :

1. Un nœud est sélectionné aléatoirement pour mise à niveau, jusqu'au maximum indisponible configuré pour le groupe de nœuds.
2. Les Pods du nœud sont vidés. Si les Pods ne quittent pas le nœud dans les 15 minutes et qu'il n'y a pas d'indicateur de force, la phase de mise à niveau échoue avec une erreur `PodEvictionFailure`. Pour ce scénario, vous pouvez appliquer l'indicateur de force avec la demande `update-nodegroup-version` de suppression des Pods.
3. Le nœud est isolé après l'expulsion de chaque Pod et un délai de 60 secondes est observé. Cela permet au contrôleur de services de ne pas envoyer de nouvelles requêtes à ce nœud et de le supprimer de sa liste de nœuds actifs.
4. Une demande d'arrêt est envoyée au groupe Auto Scaling pour le nœud isolé.
5. Les étapes de mise à niveau précédentes sont répétées jusqu'à ce que le groupe de nœuds ne comporte plus aucun nœud déployé avec la version antérieure du modèle de lancement.

Les raisons suivantes sont connues pour provoquer une erreur `PodEvictionFailure` dans cette phase :

PDB agressif

Un PDB agressif est défini sur le Pod ou il y a plusieurs PDB qui pointent vers le même Pod.

Déploiement tolérant tous les rejets

Une fois que chaque Pod est expulsé, on s'attend à ce que le nœud soit vide, car il a été rejeté lors des étapes précédentes. Toutefois, si le déploiement tolère tous les rejets, il est plus probable que le nœud ne soit pas vide, ce qui entraîne l'échec de l'expulsion du Pod.

Phase de réduction d'échelle

La phase de réduction d'échelle diminue la taille maximale et la taille souhaitée du groupe Auto Scaling d'une unité pour revenir aux valeurs avant le début de la mise à jour.

Si le flux de mise à jour détermine que le Cluster Autoscaler augmente le groupe de nœuds pendant la phase de réduction d'échelle du flux de travail, il se termine immédiatement sans ramener le groupe de nœuds à sa taille d'origine.

Rejets de nœuds sur les groupes de nœuds gérés

Amazon EKS prend en charge la configuration des rejets Kubernetes par le biais de groupes de nœuds gérés. Les rejets et les tolérances fonctionnent ensemble pour garantir que les Pods ne sont pas programmés sur des nœuds inappropriés. Un ou plusieurs rejets peuvent être appliqués à un nœud. Cette opération marque le nœud pour indiquer qu'il ne doit pas accepter de Pods qui ne tolèrent pas les rejets. Les tolérances sont appliquées aux Pods et permettent, mais n'exigent pas, que les Pods soient programmés sur des nœuds avec des rejets correspondants. Pour plus d'informations, consultez Rejets et les tolérances (français non garanti) dans la documentation de Kubernetes.

Les rejets de nœuds Kubernetes peuvent être appliqués à des groupes de nœuds gérés, nouveaux ou existants, à l'aide de l'AWS Management Console ou de l'API Amazon EKS.

- Pour plus d'informations sur la création d'un groupe de nœuds avec un rejet à l'aide de la AWS Management Console, consultez Création d'un groupe de nœuds gérés.
- Voici un exemple de création d'un groupe de nœuds avec un rejet à l'aide de l'AWS CLI :

```
aws eks create-nodegroup \
 --cli-input-json '
{
  "clusterName": "my-cluster",
  "nodegroupName": "node-taints-example",
  "subnets": [
```

```
      "subnet-1234567890abcdef0",
      "subnet-abcdef01234567890",
      "subnet-021345abcdef67890"
    ],
  "nodeRole": "arn:aws:iam::111122223333:role/AmazonEKSNodeRole",
  "taints": [
     {
         "key": "dedicated",
         "value": "gpuGroup",
         "effect": "NO_SCHEDULE"
     }
    ]
}'
```

Pour plus d'informations et des exemples d'utilisation, consultez la section concernant les rejets dans la documentation de référence de Kubernetes.

> **Note**
>
> - Les rejets peuvent être mis à jour après avoir créé le groupe de nœuds à l'aide de l'API `UpdateNodegroupConfig`.
> - La clé de rejet doit commencer par une lettre ou un chiffre. Il peut contenir des lettres, des chiffres, des traits d'union (`-`), des points (`.`) et des traits de soulignement (`_`). Il peut comporter jusqu'à 63 caractères.
> - Éventuellement, la clé de rejet peut commencer par un préfixe de sous-domaine DNS et un simple `/`. Si elle commence par un préfixe de sous-domaine DNS, elle peut comporter 253 caractères.
> - La valeur est facultative et doit commencer par une lettre ou un chiffre. Il peut contenir des lettres, des chiffres, des traits d'union (`-`), des points (`.`) et des traits de soulignement (`_`). Il peut comporter jusqu'à 63 caractères.
> - Lors de l'utilisation directe de Kubernetes ou de la AWS Management Console, l'effet du rejet doit être **`NoSchedule`**, **`PreferNoSchedule`** ou **`NoExecute`**. Toutefois, lorsque vous utilisez le AWS CLI ou l'API, l'effet du rejet doit être **`NO_SCHEDULE`**, **`PREFER_NO_SCHEDULE`** ou **`NO_EXECUTE`**.
> - Un maximum de 50 rejets est autorisé pour un groupe de nœuds.

- Si des rejets créés à l'aide d'un groupe de nœuds gérés sont supprimés manuellement d'un nœud, Amazon EKS ne rajoute pas les rejets au nœud. Ceci est vrai même si les rejets sont spécifiés dans la configuration du groupe de nœuds gérés.

Vous pouvez utiliser la commande `aws eks update-nodegroup-config` AWS CLI pour ajouter, supprimer ou remplacer des rejets pour les groupes de nœuds gérés.

Personnalisation des nœuds gérés avec des modèles de lancement

Pour le plus haut niveau de personnalisation, vous pouvez déployer des nœuds gérés en utilisant votre propre modèle de lancement. L'utilisation d'un modèle de lancement permet d'accéder à des fonctionnalités telles que les suivantes :

- Fournissez des arguments d'amorçage lors du déploiement d'un nœud, tels que des arguments `kubelet` supplémentaires.
- Affectez des adresses IP à des Pods à partir d'un bloc d'adresse CIDR différent de l'adresse IP affectée au nœud.
- Déployez votre propre AMI personnalisée sur les nœuds.
- Déployez votre propre CNI personnalisée sur les nœuds.

Si vous fournissez votre propre modèle de lancement lors de la création en premier lieu d'un groupe de nœuds gérés, vous bénéficierez également d'une plus grande flexibilité par la suite. Pourvu que vous déployiez un groupe de nœuds gérés avec votre propre modèle de lancement, vous pouvez le mettre à jour de façon itérative avec une version différente du même modèle de lancement. Lorsque vous mettez à jour votre groupe de nœuds vers une version différente de votre modèle de lancement, tous les nœuds du groupe sont recyclés pour correspondre à la nouvelle configuration de la version spécifiée du modèle de lancement.

Les groupes de nœuds gérés sont toujours déployés avec un modèle de lancement à utiliser avec le groupe Amazon EC2 Auto Scaling. Si vous ne fournissez pas de modèle de lancement, l'API Amazon EKS en crée un automatiquement avec des valeurs par défaut dans votre compte. Cependant, nous ne vous recommandons pas de modifier les modèles de lancement générés automatiquement. De plus, les groupes de nœuds existants qui n'utilisent pas de modèle de lancement personnalisé ne peuvent pas être mis à jour directement. Vous devez plutôt créer un nouveau groupe de nœuds avec un modèle de lancement personnalisé à cet effet.

Concepts de base de la configuration d'un modèle de lancement

Vous pouvez créer un modèle de lancement Amazon EC2 Auto Scaling avec le AWS Management Console AWS CLI, ou un AWS SDK. Pour plus d'informations, consultez Création d'un modèle de lancement pour un groupe Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling. Certains des paramètres d'un modèle de lancement sont similaires aux paramètres utilisés pour la configuration des nœuds gérés. Lors du déploiement ou de la mise à jour d'un groupe de nœuds avec un modèle de lancement, certains paramètres doivent être spécifiés soit dans la configuration du groupe de nœuds, soit dans le modèle de lancement. Ne spécifiez pas un paramètre aux deux endroits. Si un paramètre existe là où il ne devrait pas, des opérations telles que la création ou la mise à jour d'un groupe de nœuds échouent.

Le tableau suivant répertorie les paramètres interdits dans un modèle de lancement. Il répertorie également les paramètres similaires, s'il en existe, qui sont requis dans la configuration du groupe de nœuds géré. Les paramètres répertoriés sont les paramètres qui apparaissent dans la console. Ils peuvent avoir des noms similaires mais différents dans le SDK AWS CLI et.

Modèle de lancement – Interdit	Configuration d'un groupe de nœuds Amazon EKS
Sous-réseau sous Interfaces réseau (Ajouter une interface réseau)	Sous-réseaux sous Configuration réseau du groupe de nœuds sur la page Spécifier le réseau
Profil d'instance IAM sous Détails avancés	Rôle IAM de nœud sous Configuration du groupe de nœuds sur la page Configurer le groupe de nœuds
Comportement d'arrêt et Arrêt – Comportement de mise en veille prolongée sous Détails avancés. Rétention par défaut Ne pas inclure dans le paramètre de modèle de lancement dans le modèle de lancement pour les deux paramètres.	Pas d'équivalent. Amazon EKS doit contrôler le cycle de vie de l'instance et non pas le groupe Auto Scaling.

Le tableau suivant répertorie les paramètres interdits dans une configuration de groupe de nœuds gérée. Il répertorie également les paramètres similaires, s'il en existe, qui sont requis dans un modèle

de lancement. Les paramètres répertoriés sont les paramètres qui apparaissent dans la console. Ils peuvent porter des noms similaires dans le SDK AWS CLI et.

Configuration du groupe de nœuds Amazon EKS : Interdite	Modèle de lancement

(Seulement si vous avez spécifié une AMI personnalisée dans un modèle de lancement) AMI type (Type d'AMI) sous Node Group compute configuration (Configuration du calcul du groupe de nœuds) sur la page Set compute and scaling configuration (Définir la configuration de calcul et de mise à l'échelle) : la console affiche Specified in launch template (Spécifié dans le modèle de lancement) et l'ID d'AMI spécifié.
 	
Si aucune Image d'application et de système d'exploitation (Amazon Machine Image) n'a été spécifiée dans le modèle de lancement, vous pouvez sélectionner une AMI dans la configuration du groupe de nœuds.
	Images d'applications et de systèmes d'exploitation (Amazon Machine Image) sous Contenu du modèle de lancement : vous devez spécifier un ID dans l'un des cas suivants :
	
	• Utilisation d'une image AMI personnalisée Si vous spécifiez une AMI qui ne répond pas aux exigences énumérées dans Spécification d'une AMI, le déploiement du groupe de nœuds échouera.
	• Veut fournir des données utilisateur pour fournir des arguments au fichier `bootstrap.sh` inclus avec un AMI optimisé pour Amazon EKS. Vous pouvez permettre à vos instances d'attribuer un nombre nettement plus élevé d'adresses IPPods, d'attribuer des adresses IP à Pods partir d'un bloc CIDR différent de celui de l'instance ou de déployer un cluster privé sans accès Internet sortant. Pour plus d'informations, consultez les rubriques suivantes :
	  • Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2
	  • Mise en réseau personnalisée pour les pods
	  • Exigences relatives aux clusters privés
	  • Spécification d'une AMI

Configuration du groupe de nœuds Amazon EKS : Interdite	Modèle de lancement
Taille du disque sous Configuration de calcul du groupe de nœuds sur la page Définir la configuration de calcul et de mise à l'échelle : la console affiche Spécifié dans le modèle de lancement.	Taille sous Stockage (volumes) (Ajouter un nouveau volume). Vous devez le spécifier dans le modèle de lancement.
Paire de clés SSH sous Configuration du groupe de nœuds sur la page Spécifier le réseau : la console affiche la clé spécifiée dans le modèle de lancement ou Non spécifié dans le modèle de lancement.	Nom de la paire de clés sous Paire de clés (connexion).
Vous ne pouvez pas spécifier les groupes de sécurité source qui sont autorisés à accéder à distance lors de l'utilisation d'un modèle de lancement.	Groupes de sécurité sous Paramètres réseau pour l'instance ou Groupes de sécurité sous Interfaces réseau (Ajouter une interface réseau), mais pas les deux. Pour plus d'informations, consultez Utilisation des groupes de sécurité.

Note

- Si vous déployez un groupe de nœuds à l'aide d'un modèle de lancement, spécifiez zéro ou un type d'instance sous Contenu du modèle de lancement dans un modèle de lancement. Vous pouvez également spécifier de 0 à 20 types d'instance sous types d'instance sur la page Définir la configuration de calcul et de mise à l'échelle dans la console. Vous pouvez également le faire à l'aide d'autres outils qui utilisent l'API Amazon EKS. Si vous spécifiez un type d'instance dans un modèle de lancement et que vous utilisez ce modèle de lancement pour déployer votre groupe de nœuds, vous ne pouvez pas spécifier d'autres types d'instances dans la console ou à l'aide d'autres outils qui utilisent l'API Amazon EKS. Si vous ne spécifiez pas de type d'instance dans un modèle de lancement, dans la console ou à l'aide d'autres outils qui utilisent l'API Amazon EKS, le type d'instance `t3.medium` est utilisé. Si votre groupe de nœuds utilise le type de capacité Spot, nous vous recommandons de spécifier plusieurs types d'instance à l'aide de

la console. Pour plus d'informations, consultez Types de capacité des groupes de nœuds gérés.

- Si les conteneurs que vous déployez dans le groupe de nœuds utilisent le service de métadonnées d'instance version 2, veillez à définir 2 comme limite de saut de réponse des métadonnées dans votre modèle de lancement. Pour plus d'informations, consultez Métadonnées d'instance et données utilisateur dans le Guide de l'utilisateur Amazon EC2. Si vous déployez un groupe de nœuds géré sans utiliser de modèle de lancement personnalisé, cette valeur est automatiquement définie pour le groupe de nœuds dans le modèle de lancement par défaut.

Étiquetage des instances Amazon EC2

Vous pouvez utiliser le paramètre `TagSpecification` d'un modèle de lancement pour spécifier les identifications à appliquer aux instances Amazon EC2 dans votre groupe de nœuds. L'entité IAM appelant l'API `CreateNodegroup` ou `UpdateNodegroupVersion` doit avoir des autorisations pour `ec2:RunInstances` et `ec2:CreateTags`, et les identifications doivent être ajoutées au modèle de lancement.

Utilisation des groupes de sécurité

Vous pouvez utiliser un modèle de lancement pour spécifier des groupes de sécurité Amazon EC2 à appliquer aux instances de votre groupe de nœuds. Vous pouvez le faire soit dans le paramètre des groupes de sécurité au niveau de l'instance, soit dans le cadre des paramètres de configuration de l'interface réseau. Cependant, vous ne pouvez pas créer un modèle de lancement qui spécifie à la fois les groupes de sécurité au niveau de l'instance et de l'interface réseau. Prenez en compte les conditions suivantes qui s'appliquent à l'utilisation de groupes de sécurité personnalisés avec des groupes de nœuds gérés :

- Amazon EKS n'autorise que les modèles de lancement avec une seule spécification d'interface réseau.
- Par défaut, Amazon EKS applique le groupe de sécurité du cluster aux instances de votre groupe de nœuds pour faciliter la communication entre les nœuds et le plan de contrôle. Si vous spécifiez des groupes de sécurité personnalisés dans le modèle de lancement en utilisant l'une des options mentionnées précédemment, Amazon EKS n'ajoute pas le groupe de sécurité du cluster. Vous devez donc vous assurer que les règles d'entrée et de sortie de vos groupes de sécurité permettent la communication avec le point de terminaison de votre cluster. Si vos règles de groupe

de sécurité sont incorrectes, les composants master ne peuvent pas rejoindre le cluster. Pour plus d'informations sur les règles de groupe de sécurité, consultez Considérations et exigences relatives aux groupes de sécurité Amazon EKS.

- Si vous avez besoin d'un accès SSH aux instances de votre groupe de nœuds, incluez un groupe de sécurité qui autorise cet accès.

Données utilisateur Amazon EC2

Le modèle de lancement comprend une section pour les données utilisateur personnalisées. Vous pouvez spécifier les paramètres de configuration de votre groupe de nœuds dans cette section sans créer manuellement des AMI personnalisés individuels. Pour plus d'informations sur les paramètres disponibles pour Bottlerocket, consultez Utilisation des données utilisateur sur GitHub.

Vous pouvez fournir des données d'utilisateur Amazon EC2 dans votre modèle de lancement en utilisant `cloud-init` lors du lancement de vos instances. Pour plus d'informations, consultez la documentation cloud-init. Vos données utilisateur peuvent être utilisées pour effectuer des opérations de configuration courantes. Elles comprennent :

- Inclusion d'utilisateurs ou de groupes
- Installations de packages

Les données utilisateur Amazon EC2 dans les modèles de lancement qui sont utilisés avec les groupes de nœuds gérés doivent être au format MIME Multi-Part Archive pour les AMI Amazon Linux et au format TOML pour les AMI Bottlerocket. En effet, vos données utilisateur sont fusionnées avec les données utilisateur d'Amazon EKS requises pour que les nœuds puissent rejoindre le cluster. Ne spécifiez aucune commande dans vos données utilisateur qui démarre ou modifie `kubelet`. Ceci est effectué dans le cadre des données utilisateur fusionnées par Amazon EKS. Certains paramètres `kubelet`, tels que la définition des étiquettes sur les nœuds, peuvent être configurés directement via l'API des groupes de nœuds gérés.

> **Note**
>
> Pour plus d'informations sur la personnalisation avancée de `kubelet`, y compris son démarrage manuel ou le passage de paramètres de configuration personnalisés, consultez Spécification d'une AMI. Si un ID d'AMI personnalisé est spécifié dans un modèle de lancement, Amazon EKS ne fusionne pas les données utilisateur.

Les détails suivants fournissent plus d'informations sur la section des données utilisateur.

Amazon Linux 2 user data

Vous pouvez combiner plusieurs blocs de données utilisateur dans un seul fichier MIME multi-part. Par exemple, vous pouvez combiner un boothook de cloud qui configure le démon Docker avec un script shell de données utilisateur qui installe un package personnalisé. Un fichier MIME multi-part est constitué des composants suivants :

- La déclaration de type de contenu et de limite de partie : `Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="`
- La déclaration de version MIME : `MIME-Version: 1.0`
- Un ou plusieurs blocs de données qui contiennent les composants suivants :
  - La limite de début qui signale le début d'un bloc de données utilisateur : `--==MYBOUNDARY==`
  - La déclaration de type de contenu du bloc : `Content-Type: text/cloud-config; charset="us-ascii"`. Pour plus d'informations sur les types de contenus, consultez la documentation sur Cloud-Init.
  - Le contenu des données utilisateur (par exemple, une liste de commandes shell ou de directives `cloud-init`).
  - La limite de fin qui signale la fin du fichier MIME multi-part : `--==MYBOUNDARY==--`

Voici un exemple de fichier MIME multi-part que vous pouvez utiliser pour créer le vôtre.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo "Running custom user data script"

--==MYBOUNDARY==--
```

Amazon Linux 2023 user data

Amazon Linux 2023 (AL2023) introduit un nouveau processus d'initialisation des nœuds `nodeadm` qui utilise un schéma de configuration YAML. Si vous utilisez des groupes de nœuds autogérés ou une AMI avec un modèle de lancement, vous devez désormais fournir des métadonnées de cluster supplémentaires de manière explicite lors de la création d'un nouveau groupe de nœuds. Voici un exemple des paramètres minimaux requis, où `apiServerEndpointcertificateAuthority`, et le service `cidr` sont désormais requis :

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

Vous définissez généralement cette configuration dans vos données utilisateur, telle quelle ou intégrée dans un document MIME en plusieurs parties :

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig spec: [...]

--BOUNDARY--
```

Dans AL2, les métadonnées de ces paramètres ont été découvertes à partir de l'appel d'`DescribeCluster`API Amazon EKS. Avec AL2023, ce comportement a changé car l'appel d'API supplémentaire risque d'être limité lors de la mise à l'échelle de nœuds à grande échelle. Cette modification ne vous concerne pas si vous utilisez des groupes de nœuds gérés sans modèle de lancement ou si vous utilisezKarpenter. Pour plus d'informations sur les services

`certificateAuthority` et les services`cidr`, consultez `DescribeCluster` le manuel Amazon EKS API Reference.

Bottlerocket user data

Bottlerocket structure les données utilisateur au format TOML. Vous pouvez fournir des données utilisateur qui seront fusionnées avec les données utilisateur fournies par Amazon EKS. Par exemple, vous pouvez fournir des paramètres `kubelet` supplémentaires.

```
[settings.kubernetes.system-reserved]
cpu = "10m"
memory = "100Mi"
ephemeral-storage= "1Gi"
```

Pour plus d'informations sur les paramètres pris en charge, consultez la documentation Bottlerocket. Vous pouvez configurer des étiquettes de nœuds et des rejets dans vos données utilisateur. Cependant, nous vous recommandons de les configurer plutôt dans votre groupe de nœuds. Amazon EKS applique ces configurations lorsque vous le faites.

Lorsque les données utilisateur sont fusionnées, le formatage n'est pas préservé, mais le contenu reste le même. La configuration que vous fournissez dans vos données utilisateur remplace tous les paramètres qui sont configurés par Amazon EKS. Ainsi, si vous définissez `settings.kubernetes.max-pods` o`usettings.kubernetes.cluster-dns-ip`, ces valeurs de vos données utilisateur sont appliquées aux nœuds.

Amazon EKS ne prend pas en charge toutes les TOML valides. Voici une liste des formats connus non pris en charge :

- Guillemets dans les clés entre guillemets : `'quoted "value"' = "value"`
- Guillemets échappés dans les valeurs : `str = "I'm a string. \"You can quote me \""`
- Flottants et entiers mélangés : `numbers = [ 0.1, 0.2, 0.5, 1, 2, 5 ]`
- Types mixtes dans les tableaux : `contributors = ["foo@example.com", { name = "Baz", email = "baz@example.com" }]`
- En-têtes entre parenthèses avec clés entre guillemets : `[foo."bar.baz"]`

Windows user data

Les données utilisateur Windows utilisent des commandes PowerShell. Lorsque vous créez un groupe de nœuds gérés, vos données utilisateur personnalisées sont combinées avec les données utilisateur gérées par Amazon EKS. Vos commandes PowerShell apparaissent en premier, suivies des commandes de gestion des données utilisateur, le tout dans une seule balise **<powershell></powershell>**.

> **Note**
>
> Lorsqu'aucun ID d'AMI n'est spécifié dans le modèle de lancement, n'utilisez pas le script Windows Amazon EKS Bootstrap dans les données utilisateur pour configurer Amazon EKS.

Voici un exemple de données utilisateur.

```
<powershell>
Write-Host "Running custom user data script"
</powershell>
```

Spécification d'une AMI

Si vous avez l'une des conditions suivantes, spécifiez un ID d'AMI dans le champ `ImageId` de votre modèle de lancement. Sélectionnez votre besoin d'informations supplémentaires.

Fournir des données utilisateur pour passer des arguments au fichier **`bootstrap.sh`** inclus avec une AMI Linux/Bottlerocket optimisée pour Amazon EKS

L'amorçage est un terme utilisé pour décrire l'ajout de commandes pouvant être exécutées au démarrage d'une instance. Par exemple, l'amorçage permet d'utiliser des arguments `kubelet` supplémentaires. Vous pouvez transmettre les arguments au script `bootstrap.sh` en utilisant `eksctl` sans spécifier de modèle de lancement. Vous pouvez également le faire en spécifiant les informations dans la section des données utilisateur d'un modèle de lancement.

eksctl without specifying a launch template

Créez un fichier nommé *`my-nodegroup.yaml`* avec les contenus suivants. Remplacez chaque *`example value`* par vos propres valeurs. Les arguments `--apiserver-endpoint`, `--`

`b64-cluster-ca` et `--dns-cluster-ip` sont facultatifs. Cependant, leur définition permet au script `bootstrap.sh` d'éviter de passer un appel `describeCluster`. Ceci est utile dans les configurations de clusters privés ou dans les clusters où vous mettez à l'échelle des nœuds fréquemment. Pour plus d'informations sur le script `bootstrap.sh`, consultez le fichier `bootstrap.sh` sur GitHub.

- Le seul argument requis est le nom du cluster (*`my-cluster`*).
- Pour récupérer un ID d'AMI optimisée pour `ami-`*`1234567890abcdef0`*, vous pouvez utiliser les tables des sections suivantes :
  - Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS
  - Récupération des ID d'AMI Bottlerocket optimisées pour Amazon EKS
  - Récupération des ID d'AMI Windows optimisées pour Amazon EKS
- Pour récupérer le *`certificate-authority`* de votre cluster, exécutez la commande suivante.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Pour récupérer le *`api-server-endpoint`* de votre cluster, exécutez la commande suivante.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- La valeur de `--dns-cluster-ip` est votre CIDR de service avec `.10` à la fin. Pour récupérer le *`service-cidr`* de votre cluster, exécutez la commande suivante. Par exemple, si la valeur renvoyée est `ipv4 10.100.0.0/16`, votre valeur est *`10.100.0.10`*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- Cet exemple fournit un argument `kubelet` supplémentaire pour définir une valeur `max-pods` personnalisée à l'aide du script `bootstrap.sh` inclus dans l'AMI optimisée pour Amazon EKS. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Pour obtenir de l'aide sur la sélection de *`my-max-pods-value`*, consultez Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2.

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    instanceType: m5.large
    privateNetworking: true
    disableIMDSv1: true
    labels: { x86-al2-specified-mng }
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster \
        --b64-cluster-ca certificate-authority \
        --apiserver-endpoint api-server-endpoint \
        --dns-cluster-ip service-cidr.10 \
        --kubelet-extra-args '--max-pods=my-max-pods-value' \
        --use-max-pods false
```

Pour chaque option de fichier `eksctl config` disponible, consultez Schéma de fichier de configuration dans la documentation `eksctl`. L'utilitaire `eksctl` crée toujours un modèle de lancement pour vous et remplit ses données utilisateur avec les données que vous fournissez dans le fichier `config`.

Créez un groupe de nœuds avec la commande suivante.

```
eksctl create nodegroup --config-file=my-nodegroup.yaml
```

User data in a launch template

Spécifiez les informations suivantes dans la section des données utilisateur de votre modèle de lancement. Remplacez chaque *`example value`* par vos propres valeurs. Les arguments `--apiserver-endpoint`, `--b64-cluster-ca` et `--dns-cluster-ip` sont facultatifs. Cependant, leur définition permet au script `bootstrap.sh` d'éviter de passer un appel `describeCluster`. Ceci est utile dans les configurations de clusters privés ou dans les clusters

où vous mettez à l'échelle des nœuds fréquemment. Pour plus d'informations sur le script `bootstrap.sh`, consultez le fichier bootstrap.sh sur GitHub.

- Le seul argument requis est le nom du cluster (*my-cluster*).
- Pour récupérer le *certificate-authority* de votre cluster, exécutez la commande suivante.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Pour récupérer le *api-server-endpoint* de votre cluster, exécutez la commande suivante.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- La valeur de `--dns-cluster-ip` est votre CIDR de service avec `.10` à la fin. Pour récupérer le *service-cidr* de votre cluster, exécutez la commande suivante. Par exemple, si la valeur renvoyée est `ipv4 10.100.0.0/16`, votre valeur est *10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- Cet exemple fournit un argument `kubelet` supplémentaire pour définir une valeur `max-pods` personnalisée à l'aide du script `bootstrap.sh` inclus dans l'AMI optimisée pour Amazon EKS. Pour obtenir de l'aide sur la sélection de *my-max-pods-value*, consultez Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
set -ex
/etc/eks/bootstrap.sh my-cluster \
  --b64-cluster-ca certificate-authority \
  --apiserver-endpoint api-server-endpoint \
  --dns-cluster-ip service-cidr.10 \
  --kubelet-extra-args '--max-pods=my-max-pods-value' \
```

```
  --use-max-pods false

--==MYBOUNDARY==--
```

Fournir des données utilisateur pour passer des arguments au fichier **`Start-EKSBootstrap.ps1`** inclus avec une AMI Windows optimisée pour Amazon EKS

L'amorçage est un terme utilisé pour décrire l'ajout de commandes pouvant être exécutées au démarrage d'une instance. Vous pouvez transmettre les arguments au script `Start-EKSBootstrap.ps1` en utilisant `eksctl` sans spécifier de modèle de lancement. Vous pouvez également le faire en spécifiant les informations dans la section des données utilisateur d'un modèle de lancement.

Si vous souhaitez spécifiez un ID d'AMI Windows personnalisée, gardez à l'esprit les considérations suivantes :

- Vous devez utiliser un modèle de lancement et fournir les commandes d'amorçage requises dans la section des données utilisateur. Pour récupérer l'ID Windows souhaité, vous pouvez utiliser le tableau dans AMI Windows optimisées pour Amazon EKS.
- Il existe plusieurs limites et conditions. Par exemple, vous devez ajouter des éléments `eks:kube-proxy-windows` à votre carte de configuration AWS IAM Authenticator. Pour plus d'informations, consultez Limites et conditions lors de la spécification d'un ID d'AMI.

Spécifiez les informations suivantes dans la section des données utilisateur de votre modèle de lancement. Remplacez chaque *`example value`* par vos propres valeurs. Les arguments `-APIServerEndpoint`, `-Base64ClusterCA` et `-DNSClusterIP` sont facultatifs. Cependant, leur définition permet au script `Start-EKSBootstrap.ps1` d'éviter de passer un appel `describeCluster`.

- Le seul argument requis est le nom du cluster (*`my-cluster`*).
- Pour récupérer le *`certificate-authority`* de votre cluster, exécutez la commande suivante.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text --
name my-cluster --region region-code
```

- Pour récupérer le *`api-server-endpoint`* de votre cluster, exécutez la commande suivante.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-cluster
 --region region-code
```

- La valeur de `--dns-cluster-ip` est votre CIDR de service avec `.10` à la fin. Pour récupérer le *`service-cidr`* de votre cluster, exécutez la commande suivante. Par exemple, si la valeur renvoyée est `ipv4 10.100.0.0/16`, votre valeur est *`10.100.0.10`*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr" --
output text --name my-cluster --region region-code
```

- Pour des arguments supplémentaires, consultez Paramètres de configuration du script d'amorçage.

> **Note**
>
> Si vous utilisez un CIDR de service personnalisé, vous devez le spécifier à l'aide du paramètre `-ServiceCIDR`. Dans le cas contraire, la résolution DNS pour les Pods dans le cluster échouera.

```
<powershell>
[string]$EKSBootstrapScriptFile = "$env:ProgramFiles\Amazon\EKS\Start-EKSBootstrap.ps1"
& $EKSBootstrapScriptFile -EKSClusterName my-cluster `
  -Base64ClusterCA certificate-authority `
  -APIServerEndpoint api-server-endpoint `
  -DNSClusterIP service-cidr.10
</powershell>
```

Exécution d'une AMI personnalisée en raison d'exigences spécifiques en matière de sécurité, de conformité ou de politique interne.

Pour plus d'informations, veuillez consulter la rubrique Amazon Machine Images (AMI) dans le Guide de l'utilisateur Amazon EC2. La spécification de génération de l'AMI Amazon EKS contient des ressources et des scripts de configuration pour créer une AMI Amazon EKS personnalisée basée sur Amazon Linux. Pour plus d'informations, consultez Spécifications de création d'AMI Amazon EKS sur GitHub. Pour créer des AMI personnalisées installées avec d'autres systèmes d'exploitation, consultez Exemples d'AMI personnalisées Amazon EKS sur GitHub.

> **Important**
>
> Lorsque vous spécifiez une AMI, Amazon EKS ne fusionne aucune donnée utilisateur. Vous devez fournir les commandes `bootstrap` requises pour que les nœuds rejoignent le cluster. Si vos nœuds ne parviennent pas à joindre le cluster, les actions `CreateNodegroup` et `UpdateNodegroupVersion` Amazon EKS échouent également.

Limites et conditions lors de la spécification d'un ID d'AMI

Voici les limites et les conditions impliquées dans la spécification d'un ID d'AMI avec des groupes de nœuds gérés :

- Vous devez créer un groupe de nœuds pour passer de la spécification d'un ID d'AMI dans un modèle de lancement à la non-spécification d'un ID d'AMI.
- Vous n'êtes pas averti dans la console lorsqu'une version plus récente de l'AMI est disponible. Pour mettre à jour votre groupe de nœuds vers une version d'AMI plus récente, vous devez créer une nouvelle version de votre modèle de lancement avec un ID d'AMI mis à jour. Vous devez ensuite mettre à jour le groupe de nœuds avec la nouvelle version du modèle de lancement.
- Les champs suivants ne peuvent pas être définis dans l'API si vous spécifiez un ID d'AMI :
  - `amiType`
  - `releaseVersion`
  - `version`
- Tous les `taints` définis dans l'API sont appliqués de manière asynchrone si vous spécifiez un identifiant d'AMI. Pour appliquer des rejets avant l'ajout d'un nœud au cluster, vous devez les transférer à `kubelet` dans vos données utilisateur à l'aide de l'indicateur de ligne de commande `--register-with-taints`. Pour plus d'informations, consultez la section `kubelet`dans la documentation Kubernetes.
- Lorsque vous spécifiez un ID AMI personnalisé pour les groupes de nœuds Windows gérés, ajoutez-le `eks:kube-proxy-windows` à votre carte de configuration AWS IAM Authenticator. Cela est nécessaire pour le bon fonctionnement du DNS.

  1. Ouvrez la carte de configuration de l'authentificateur AWS IAM pour la modifier.

     ```
     kubectl edit -n kube-system cm aws-auth
     ```

2. Ajoutez cette entrée à la liste des `groups` sous chaque `rolearn` associé aux nœuds Windows. Votre mappage de configuration doit ressembler à `aws-auth-cm-windows.yaml`.

```
- eks:kube-proxy-windows
```

3. Enregistrez le fichier et quittez votre éditeur de texte.

Suppression d'un groupe de nœuds gérés

Cette rubrique explique comment supprimer un groupe de nœuds gérés Amazon EKS. Lorsque vous supprimez un groupe de nœuds gérés, Amazon EKS définit d'abord 0 pour les tailles minimale, maximale et souhaitée de votre groupe Auto Scaling. Cela entraîne ensuite une réduction d'échelle de votre groupe de nœuds.

Avant de résilier chaque instance, Amazon EKS envoie un signal pour purger les Pods de ce nœud. Si les Pods ne se sont pas purgés après quelques minutes, Amazon EKS laisse Auto Scaling poursuivre la résiliation de l'instance. Après la résiliation de chaque instance, le groupe Auto Scaling est supprimé.

> **Important**
>
> Si vous supprimez un groupe de nœuds gérés qui utilise un rôle IAM de nœud qui n'est utilisé par aucun autre groupe de nœuds gérés dans le cluster, le rôle est supprimé de la `ConfigMap aws-auth`. Si l'un des groupes de nœuds autogérés dans le cluster utilisent le même rôle IAM de nœud, le statut des nœuds autogérés devient `NotReady`. De plus, le fonctionnement du cluster est également perturbé. Pour ajouter un mappage pour le rôle que vous utilisez uniquement pour les groupes de nœuds autogérés, consultez Création d'entrées d'accès si la version de la plateforme de votre cluster est au moins la version minimale répertoriée dans la section des prérequis de Gérer les entrées d'accès. Si la version de votre plateforme est antérieure à la version minimale requise pour les entrées d'accès, vous pouvez réajouter l'entrée à la `ConfigMap aws-auth`. Pour plus d'informations, entrez `eksctl create iamidentitymapping --help` dans votre terminal.

Vous pouvez supprimer un groupe de nœuds gérés avec `eksctl` ou la AWS Management Console.

eksctl

Pour supprimer un groupe de nœuds gérés avec **`eksctl`**

Entrez la commande suivante. Remplacez chaque *`example value`* par vos propres valeurs.

```
eksctl delete nodegroup \
  --cluster my-cluster \
  --name my-mng \
  --region region-code
```

Pour plus d'options, consultez la section Supprimer et vider des groupes de nœuds dans la documentation `eksctl`.

AWS Management Console

Pour supprimer votre groupe de nœuds géré à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Sur la page Clusters, choisissez le cluster qui contient le groupe de nœuds à supprimer.
3. Sur la page du cluster sélectionné, choisissez l'onglet Compute.
4. Dans la section Node Groups (Groupes de nœuds), choisissez le groupe de nœuds à supprimer. Ensuite, choisissez Supprimer.
5. Dans la boîte de dialogue de confirmation de la suppression du groupe de nœuds, entrez le nom du groupe de nœuds. Ensuite, choisissez Supprimer.

AWS CLI

Pour supprimer votre groupe de nœuds géré à l'aide du AWS CLI

1. Entrez la commande suivante. Remplacez chaque *`example value`* par vos propres valeurs.

   ```
   aws eks delete-nodegroup \
     --cluster-name my-cluster \
     --nodegroup-name my-mng \
     --region region-code
   ```

2. Utilisez les touches fléchées de votre clavier pour faire défiler la sortie de réponse. Appuyez sur la touche **q** lorsque vous avez terminé.

Pour plus d'options, consultez la commande `delete-nodegroup` dans la section Référence des commandes AWS CLI .

Nœuds autogérés

Un cluster contient un ou plusieurs nœuds Amazon EC2 sur lesquels les Pods sont programmés. Les nœuds Amazon EKS s'exécutent dans votre AWS compte et se connectent au plan de contrôle de votre cluster via le point de terminaison du serveur API du cluster. Ils vous sont facturés en fonction des tarifs d'Amazon EC2. Pour plus d'informations, consultez Tarification Amazon EC2.

Un cluster peut contenir plusieurs groupes de nœuds. Chaque groupe de nœuds contient un ou plusieurs nœuds déployés dans un groupe Amazon EC2 Auto Scaling. Le type d'instance des nœuds du groupe peut varier, par exemple lors de l'utilisation de la sélection du type d'instance basée sur les attributs avec Karpenter. Toutes les instances d'un groupe de nœuds doivent utiliser le rôle IAM du nœud Amazon EKS.

Amazon EKS fournit des images Amazon Machine Image (AMI) spécialisées appelées « AMI optimisées pour Amazon EKS ». Les AMI sont configurées pour fonctionner avec Amazon EKS. Leurs composants incluent`containerd`,`kubelet`, et l'authentificateur AWS IAM. Les AMI contiennent également un script d'amorçage spécialisé qui lui permet d'identifier le plan de contrôle de votre cluster et de s'y connecter automatiquement.

Si vous limitez l'accès au point de terminaison public de votre cluster à l'aide de blocs CIDR, nous vous recommandons d'activer également l'accès au point de terminaison privé. Cela permet aux nœuds de communiquer avec le cluster. Sans l'activation du point de terminaison privé, les blocs CIDR que vous spécifiez pour l'accès public doivent inclure les sources de sortie de votre VPC. Pour plus d'informations, consultez Contrôle d'accès au point de terminaison du cluster Amazon EKS..

Pour ajouter des nœuds autogérés à votre cluster Amazon EKS, consultez les rubriques qui suivent. Si vous lancez des nœuds autogérés manuellement, ajoutez l'identification suivante à chaque nœud. Pour de plus amples informations, veuillez consulter Ajout et suppression de balises sur une ressource individuelle. Si vous suivez les étapes des guides qui suivent, l'identification requise est ajoutée au nœud pour vous.

Clé	Valeur
`kubernetes.io/cluster/ my-cluster`	`owned`

Pour plus d'informations sur les nœuds à partir d'une perspective Kubernetes générale, consultez Nodes (Nœuds) dans la documentation Kubernetes.

Rubriques

- Lancement de nœuds Amazon Linux autogérés
- Lancement de nœuds Bottlerocket autogérés
- Lancement de nœuds Windows autogérés
- Lancement de nœuds Ubuntu autogérés
- Mises à jour des nœuds autogérés

Lancement de nœuds Amazon Linux autogérés

Cette rubrique décrit comment lancer des groupes Auto Scaling de nœuds Linux qui s'enregistrent auprès de votre cluster Amazon EKS. Une fois que les nœuds ont rejoint le cluster, vous pouvez y déployer des applications Kubernetes. Vous pouvez également lancer des nœuds Amazon Linux autogérés avec `eksctl` ou le AWS Management Console. Si vous devez lancer des nœuds sur AWS Outposts, consultezLancement de nœuds Amazon Linux autogérés sur un Outpost.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Création d'un cluster Amazon EKS. Si vous avez des sous-réseaux Région AWS là où vous l'avez activé AWS Outposts AWS Wavelength, ou si vous avez activé AWS des Zones Locales, ces sous-réseaux ne doivent pas avoir été transmis lorsque vous avez créé votre cluster.
- Un rôle IAM existant pour les nœuds à utiliser. Pour en créer un, consultez Rôle IAM de nœud Amazon EKS. Si ce rôle ne comporte aucune des politiques pour le VPC CNI, le rôle distinct suivant est nécessaire pour les pods VPC CNI.
- (Facultatif, mais recommandé) Le module complémentaire Amazon VPC CNI plugin for Kubernetes configuré avec son propre rôle IAM auquel est attachée la politique IAM nécessaire. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
- Être familiarisé avec les considérations énumérées dans Choix d'un type d'instance Amazon EC2. Selon le type d'instance que vous choisissez, il peut y avoir des prérequis supplémentaires pour votre cluster et votre VPC.

eksctl

> **Note**
>
> `eksctl`n'est pas compatible avec Amazon Linux 2023 pour le moment.

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour lancer des nœuds Linux autogérés à l'aide de **`eksctl`**

1. (Facultatif) Si la politique d'IAM gérée AmazonEKS_CNI_Policy est associée à votre Rôle IAM de nœud Amazon EKS, nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service Kubernetes `aws-node` à la place. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
2. La commande suivante crée un groupe de nœuds dans un cluster existant. Remplacer *`al-nodes`* avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Remplacez *`my-cluster`* par le nom de votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Remplacez les valeurs de *`example value`* restantes par vos propres valeurs. Par défaut, les nœuds sont créés avec la même version de Kubernetes que le plan de contrôle.

   Avant de choisir une valeur pour `--node-type`, examinez Choix d'un type d'instance Amazon EC2.

   Remplacez *`my-key`* par le nom de votre paire de clés Amazon EC2 ou de votre clé publique. Cette clé est utilisée pour SSH dans vos nœuds après leur lancement. Si vous ne possédez pas d'une paire de clés Amazon EC2, vous pouvez en créer une dans la AWS Management Console. Pour plus d'informations, veuillez consulter la rubrique Paires de clés Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Créez votre groupe de nœuds avec la commande suivante.

> **Important**
>
> Si vous souhaitez déployer un groupe de nœuds sur des sous-réseaux AWS Outposts Wavelength ou Local Zone, vous devez prendre en compte d'autres considérations :
>
> - Les sous-réseaux ne doivent pas avoir été renseignés lors de la création du cluster.
> - Vous devez créer le groupe de nœuds avec un fichier config qui spécifie les sous-réseaux et `volumeType: gp2`. Pour plus d'informations, consultez Créer un nodegroup à partir d'un fichier config et Schéma du fichier Config dans la documentation `eksctl`.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --name al-nodes \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --ssh-access \
  --managed=false \
  --ssh-public-key my-key
```

Pour déployer un groupe de nœuds qui :

- peut attribuer un nombre nettement plus élevé d'adresses IP à Pods comparé à la configuration par défaut, veuillez consulter Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.
- peut assigner les adresses de `IPv4` à Podsd'un autre bloc CIDR supérieur à l'instance, consultez Mise en réseau personnalisée pour les pods.
- peut attribuer des adresses `IPv6` aux Pods et services, veuillez consulter la rubrique `IPv6`adresses pour les clustersPods, et services.

- Pour utiliser l'environnement d'exécution `containerd`, vous devez déployer le groupe de nœuds à l'aide d'un fichier `config`. Pour plus d'informations, consultez Testez la migration Docker de `containerd`.
- n'avez pas d'accès Internet sortant, veuillez consulter Exigences relatives aux clusters privés.

Pour obtenir la liste complète de toutes les options disponibles et des valeurs par défaut, saisissez la commande suivante.

```
eksctl create nodegroup --help
```

Si les nœuds ne parviennent pas à rejoindre le cluster, reportez-vous à Les nœuds ne parviennent pas à joindre le cluster dans le guide de dépannage.

L'exemple qui suit illustre un résultat. Plusieurs lignes sont affichées pendant la création des nœuds. L'une des dernières lignes de sortie est similaire à la ligne d'exemple suivante.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facultatif) Déployez un exemple d'application pour tester votre cluster et les nœuds Linux.
4. Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

   - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
   - Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

   Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

AWS Management Console

Étape 1 : pour lancer des nœuds Linux autogérés à l'aide de AWS Management Console

1. Téléchargez la dernière version du AWS CloudFormation modèle.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
   ```

2. Attendez que le statut de votre cluster s'affiche soit `ACTIVE`. Si vous lancez vos nœuds avant que le cluster soit actif, les nœuds ne s'enregistrent pas avec le cluster et vous devrez les relancer.
3. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
4. Choisissez Create stack (Créer une pile), puis sélectionnez Avec de nouvelles ressources (standard).
5. Pour Spécifier un modèle, sélectionnez Upload a template file (Télécharger un fichier de modèle), puis sélectionnez Choose file (Choisir un fichier).
6. Sélectionnez le fichier `amazon-eks-nodegroup.yaml` que vous avez téléchargé.
7. Sélectionnez Suivant.
8. Dans la page Specify stack details (Spécifier les détails de la pile), saisissez les paramètres suivants, puis choisissez Next (Suivant) :

   - Nom de la pile : choisissez un nom pour votre pile AWS CloudFormation . Par exemple, vous pouvez l'appeler ***my-cluster-nodes***. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
   - ClusterName: Entrez le nom que vous avez utilisé lors de la création de votre cluster Amazon EKS. Ce nom doit être égal au nom du cluster, sinon vos nœuds ne pourront pas rejoindre le cluster.
   - ClusterControlPlaneSecurityGroupe : Choisissez la SecurityGroupsvaleur à partir de la AWS CloudFormation sortie que vous avez générée lors de la création de votre VPC.

     Les étapes suivantes montrent une opération permettant de récupérer le groupe applicable.

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster.
3. Choisissez l'onglet Networking (Mise en réseau).
4. Utilisez la valeur Groupes de sécurité supplémentaires comme référence lorsque vous effectuez une sélection dans la liste déroulante des ClusterControlPlaneSecuritygroupes.

- NodeGroupNom : entrez le nom de votre groupe de nœuds. Ce nom peut être utilisé ultérieurement pour identifier le groupe de nœuds Auto Scaling qui est créé pour vos nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.
- NodeAutoScalingGroupMinSize: Entrez le nombre minimum de nœuds que votre groupe Auto Scaling de nœuds peut atteindre.
- NodeAutoScalingGroupDesiredCapacity: Entrez le nombre de nœuds que vous souhaitez atteindre lors de la création de votre pile.
- NodeAutoScalingGroupMaxSize: Entrez le nombre maximum de nœuds que votre groupe Auto Scaling de nœuds peut atteindre.
- NodeInstanceType : Choisissez un type d'instance pour vos nœuds. Pour plus d'informations, consultez Choix d'un type d'instance Amazon EC2.
- NodeImageIDSSMParam : prérempli avec le paramètre Amazon EC2 Systems Manager d'une AMI récemment optimisée pour Amazon EKS pour une version variable. Kubernetes Pour utiliser une autre version mineure de Kubernetes prise en charge avec Amazon EKS, remplacez *1.XX* par une autre version prise en charge. Nous vous recommandons de spécifier la même version de Kubernetes que celle de votre cluster.

  Vous pouvez également le remplacer *amazon-linux-2* par un autre type d'AMI. Pour plus d'informations, consultez Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS.

  > **Note**
  >
  > L'AMI du nœud Amazon EKS est basée sur Amazon Linux. Vous pouvez suivre les évènements de sécurité et de confidentialité pour Amazon Linux 2 via le centre de sécurité Amazon Linux ou souscrire au flux RSS associé. Les événements de

sécurité et de confidentialité incluent une présentation du problème, les packages concernés et la manière de mettre à jour vos instances pour résoudre le problème.

- NodeImageID : (Facultatif) Si vous utilisez votre propre AMI personnalisée (au lieu de l'AMI optimisée pour Amazon EKS), entrez un ID d'AMI de nœud pour votre Région AWS. Si vous spécifiez une valeur ici, elle remplace toutes les valeurs du champ NodeImageIDSSMParam.
- NodeVolumeTaille : Spécifiez une taille de volume racine pour vos nœuds, en GiB.
- NodeVolumeType : Spécifiez un type de volume racine pour vos nœuds.
- KeyName: Entrez le nom d'une paire de clés SSH Amazon EC2 que vous pourrez utiliser pour vous connecter via SSH à vos nœuds après leur lancement. Si vous ne possédez pas déjà une paire de clés Amazon EC2, vous pouvez en créer une dans l' AWS Management Console. Pour plus d'informations, veuillez consulter la rubrique Paires de clés Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

  > **Note**
  >
  > Si vous ne fournissez pas de paire de clés ici, la création de la AWS CloudFormation pile échoue.

- BootstrapArguments: Spécifiez tous les arguments facultatifs à transmettre au script bootstrap du nœud, tels que des `kubelet` arguments supplémentaires. Pour de plus amples informations, veuillez consulter Utilisation du script d'amorçage sur GitHub.

  Pour déployer un groupe de nœuds qui :

  - peut attribuer un nombre nettement plus élevé d'adresses IP à Pods comparé à la configuration par défaut, veuillez consulter Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.
  - peut assigner les adresses de `IPv4` à Podsd'un autre bloc CIDR supérieur à l'instance, consultez Mise en réseau personnalisée pour les pods.
  - peut attribuer des adresses `IPv6` aux Pods et services, veuillez consulter la rubrique `IPv6`adresses pour les clustersPods, et services.
  - Pour utiliser l'environnement d'exécution `containerd`, vous devez déployer le groupe de nœuds à l'aide d'un fichier `config`. Pour plus d'informations, consultez Testez la migration Docker de `containerd`.

  - n'avez pas d'accès Internet sortant, veuillez consulter Exigences relatives aux clusters privés.
- DisableIMDSv1 : par défaut, chaque nœud prend en charge le service de métadonnées d'instance version 1 (IMDSv1) et IMDSv2. Vous pouvez désactiver IMDSv1. Pour empêcher les futurs nœuds et Pods du groupe de nœuds d'utiliser MDSv1, définissez DisableIMDSv1 sur true (vrai). Pour de plus amples informations au sujet d'IMDS, consultez Configuration du service des métadonnées d'instance. Pour plus d'informations sur la façon d'en restreindre l'accès sur vos nœuds, consultez Restreindre l'accès au profil d'instance affecté au composant master.
- VpcId: Entrez l'ID du VPC que vous avez créé.
- Sous-réseaux : choisissez les sous-réseaux que vous avez créés pour votre VPC. Si vous avez créé votre VPC à l'aide des étapes décrites dans Création d'un VPC pour votre cluster Amazon EKS, spécifiez uniquement les sous-réseaux privés du VPC dans lesquels vos nœuds seront lancés. Vous pouvez consulter les sous-réseaux privés en ouvrant le lien de chaque sous-réseau depuis l'onglet Networking (Mise en réseau) de votre cluster.

  ⚠ Important

  - Si certains sous-réseaux sont des sous-réseaux publics, leur paramètre d'attribution automatique d'adresse IP publique doit être activé. Si le paramètre n'est pas activé pour le sous-réseau public, aucun nœud que vous déployez sur ce sous-réseau public ne se verra attribuer d'adresse IP publique et ne pourra pas communiquer avec le cluster ou d'autres AWS services. Si le sous-réseau a été déployé avant le 26 mars 2020 en utilisant l'un des modèles de AWS CloudFormation VPC Amazon EKS ou `eksctl` en utilisant, l'attribution automatique d'adresses IP publiques est désactivée pour les sous-réseaux publics. Pour plus d'informations sur l'activation de l'attribution d'adresse IP publique pour un sous-réseau, consultez Modification de l'attribut d'adressage `IPv4` public de votre sous-réseau. Si le nœud est déployé sur un sous-réseau privé, il est capable de communiquer avec le cluster et d'autres AWS services via une passerelle NAT.
  - Si les sous-réseaux ne disposent pas d'un accès Internet, assurez-vous de connaître toutes les considérations et les étapes supplémentaires indiqués dans Exigences relatives aux clusters privés.

- Si vous sélectionnez AWS Outposts des sous-réseaux Wavelength ou Local Zone, les sous-réseaux ne doivent pas avoir été transmis lors de la création du cluster.

9. Sélectionnez les choix que vous souhaitez sur la page Configure stack options (Configurer les options de la pile), puis choisissez Next (Suivant).
10. Cochez la case à gauche de Je comprends que AWS CloudFormation pourrait créer des ressources IAM., puis choisissez Create stack (Créer une pile).
11. Lorsque la création de votre pile est terminée, sélectionnez la pile dans la console et choisissez Outputs (Sorties).
12. Enregistrez le NodeInstancerôle du groupe de nœuds créé. Vous en aurez besoin lors de la configuration de vos nœuds pour Amazon EKS.

Étape 2 : pour autoriser les nœuds à rejoindre votre cluster

> **Note**
>
> Si vous avez lancé des nœuds dans un VPC privé sans accès Internet sortant, assurez-vous d'activer les nœuds pour rejoindre votre cluster à partir du VPC.

1. Vérifiez si vous avez déjà appliqué le `ConfigMap` `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si vous voyez un `ConfigMap` `aws-auth`, mettez-le à jour si nécessaire.

   a. Ouvrez le `ConfigMap` pour le modifier.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Ajoutez une nouvelle entrée `mapRoles` si nécessaire. Définissez la `rolearn` valeur sur la valeur du NodeInstancerôle que vous avez enregistrée lors de la procédure précédente.

```
[...]
data:
  mapRoles: |
```

```
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

c. Enregistrez le fichier et quittez votre éditeur de texte.

3. Si vous avez reçu un message d'erreur indiquant « `Error from server (NotFound): configmaps "aws-auth" not found` », appliquez le stock `ConfigMap`.

a. Téléchargez la mappe de configuration.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/aws-auth-cm.yaml
```

b. Dans le `aws-auth-cm.yaml` fichier, définissez la `rolearn` valeur sur la valeur du NodeInstancerôle que vous avez enregistrée lors de la procédure précédente. Pour ce faire, utilisez un éditeur de texte ou remplacez *`my-node-instance-role`* et exécutez la commande suivante :

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
```

c. Appliquez la configuration. L'exécution de cette commande peut prendre quelques minutes.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Observez le statut de vos nœuds et attendez qu'ils obtiennent le statut `Ready`.

```
kubectl get nodes --watch
```

Saisissez `Ctrl+C` pour revenir à une invite de shell.

> **Note**
>
> Si vous recevez d'autres erreurs concernant les types d'autorisations ou de ressources, consultez Accès non autorisé ou refusé (`kubectl`) dans la rubrique relative à la résolution des problèmes.

Si les nœuds ne parviennent pas à rejoindre le cluster, reportez-vous à Les nœuds ne parviennent pas à joindre le cluster dans le guide de dépannage.

5. (Nœuds GPU uniquement) Si vous avez opté pour une instance de type GPU et l'AMI accélérée optimisée pour Amazon EKS, vous devez mettre en œuvre le plugin de périphérique NVIDIA pour Kubernetes en tant que DaemonSet sur votre cluster. Remplacez *`vX.X.X`* par la version NVIDIA/k8s-device-plugin souhaitée avant d'exécuter la commande suivante.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
```

Étape 3 : actions supplémentaires

1. (Facultatif) Déployez un exemple d'application pour tester votre cluster et les nœuds Linux.
2. (Facultatif) Si la politique IAM gérée AmazonEKS_CNI_Policy (si vous avez un cluster `IPv4`) ou la politique *`AmazonEKS_CNI_IPv6_Policy`* (que vous avez créée vous-même si vous avez un cluster `IPv6`) est associée à votre the section called “Rôle IAM de nœud”, nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service `aws-node` de Kubernetes à la place. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
3. Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

   - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.

- Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Blocs de capacité pour ML

> **Important**
>
> - Les blocs de capacité ne sont disponibles que pour certains types d'instances Amazon EC2 et. Régions AWS Pour plus d'informations sur la compatibilité, consultez les prérequis relatifs à l'utilisation des blocs de capacité dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.
> - Les blocs de capacité ne peuvent actuellement pas être utilisés avec les groupes de nœuds gérés par Amazon EKS ouKarpenter.

Les blocs de capacité pour le Machine Learning (ML) vous permettent de réserver des instances GPU à une date ultérieure pour prendre en charge vos charges de travail de ML de courte durée. Les instances qui s'exécutent dans un bloc de capacité sont automatiquement placées à proximité les unes des autres dans Amazon EC2 UltraClusters, il n'est donc pas nécessaire d'utiliser un groupe de placement de clusters. Pour plus d'informations, consultez Capacity Blocks for ML dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.

Vous pouvez utiliser les blocs de capacité avec Amazon EKS pour l'approvisionnement et la mise à l'échelle de vos nœuds autogérés. Les étapes suivantes vous donnent un aperçu général de l'exemple.

1. Créez un modèle de lancement dans la AWS Management Console. Pour plus d'informations, consultez la section Utiliser les blocs de capacité pour les charges de travail d'apprentissage automatique dans le guide de l'utilisateur Amazon EC2 Auto Scaling.

   Assurez-vous d'inclure la configuration du type d'instance et de Amazon Machine Image (AMI).
2. Liez le bloc de capacité à un modèle de lancement à l'aide de l'identifiant de réserve de capacité.

Voici un exemple de AWS CloudFormation modèle pour créer un modèle de lancement ciblant un bloc de capacité :

```
NodeLaunchTemplate:
    Type: "AWS::EC2::LaunchTemplate"
    Properties:
      LaunchTemplateData:
        InstanceMarketOptions:
          MarketType: "capacity-block"
        CapacityReservationSpecification:
          CapacityReservationTarget:
            CapacityReservationId: "cr-02168da1478b509e0"
        IamInstanceProfile:
          Arn: iam-instance-profile-arn
        ImageId: image-id
        InstanceType: p5.48xlarge
        KeyName: key-name
        SecurityGroupIds:
        - sg-05b1d815d1EXAMPLE
        UserData: user-data
```

Vous devez transmettre le sous-réseau de la zone de disponibilité dans laquelle la réservation est effectuée, car les blocs de capacité sont répartis par zone.

3. Si vous créez le groupe de nœuds autogérés avant que la réserve de capacité ne soit active, définissez la capacité souhaitée sur `0`. Lorsque vous créez le groupe de nœuds, assurez-vous que vous ne spécifiez que le sous-réseau correspondant à la zone de disponibilité dans laquelle la capacité est réservée.

   Voici un exemple de CloudFormation modèle qui peut être utilisé. Cet exemple permet d'obtenir le `LaunchTemplateId` et `Version` de la ressource `AWS::Amazon EC2::LaunchTemplate` présentée dans l'exemple précédent. Il obtient également les valeurs pour `DesiredCapacity`, `MaxSize`, `MinSize`, et `VPCZoneIdentifier` qui sont déclarées ailleurs dans le même modèle.

   ```
   NodeGroup:
     Type: "AWS::AutoScaling::AutoScalingGroup"
     Properties:
       DesiredCapacity: !Ref NodeAutoScalingGroupDesiredCapacity
       LaunchTemplate:
         LaunchTemplateId: !Ref NodeLaunchTemplate
   ```

```
        Version: !GetAtt NodeLaunchTemplate.LatestVersionNumber
      MaxSize: !Ref NodeAutoScalingGroupMaxSize
      MinSize: !Ref NodeAutoScalingGroupMinSize
      VPCZoneIdentifier: !Ref Subnets
      Tags:
        - Key: Name
          PropagateAtLaunch: true
          Value: !Sub ${ClusterName}-${NodeGroupName}-Node
        - Key: !Sub kubernetes.io/cluster/${ClusterName}
          PropagateAtLaunch: true
          Value: owned
```

4. Une fois que le groupe de nœuds a été créé avec succès, assurez-vous d'enregistrer le `NodeInstanceRole` pour le groupe de nœuds qui a été créé. Vous en avez besoin pour vous assurer que lorsque le groupe de nœuds sera redimensionné, les nouveaux nœuds rejoindront le cluster et Kubernetes pourra reconnaître les nœuds. Pour plus d'informations, consultez les instructions de la AWS Management Console figurant dans Lancement de nœuds Amazon Linux autogérés.
5. Nous vous recommandons de créer une politique de mise à l'échelle programmée pour le groupe Auto Scaling qui s'aligne sur les heures de réserve du bloc de capacité. Pour plus d'informations, consultez la section Mise à l'échelle programmée pour Amazon EC2 Auto Scaling dans le Guide de l'utilisateur d'Amazon EC2 Auto Scaling.

   Vous pouvez utiliser toutes les instances que vous avez réservées jusqu'à 30 minutes avant l'heure de fin du bloc de capacité. Les instances encore en cours d'exécution à ce moment-là commenceront à prendre fin. Afin de disposer de suffisamment de temps pour vidanger le(s) nœud(s) de manière gracieuse, nous vous conseillons de programmer une mise à l'échelle jusqu'à zéro plus de 30 minutes avant l'heure de fin de la réserve du bloc de capacité.

   Si, au lieu de cela, vous souhaitez augmenter manuellement la capacité chaque fois que la réserve de capacité devient `Active`, vous devez mettre à jour la capacité souhaitée du groupe Auto Scaling à l'heure de début de la réserve du bloc de capacité. Ensuite, vous devrez également réduire manuellement la capacité plus de 30 minutes avant l'heure de fin de la réserve du bloc de capacité.
6. Le groupe de nœuds est maintenant prêt à recevoir les charges de travail et à programmer Pods.
7. Pour que votre système soit Pods correctement vidé, nous vous recommandons de configurer AWS Node Termination Handler. Ce gestionnaire sera en mesure de surveiller les événements du cycle de vie « ASG Scale-in » liés à Amazon EC2 Auto Scaling en utilisant EventBridge et de laisser le plan de Kubernetes contrôle prendre les mesures nécessaires avant que l'instance ne

soit indisponible. Dans le cas contraire, vos Pods et objets Kubernetes resteront bloqués dans un état d'attente. Pour plus d'informations, consultez AWS Node Termination Handler activé GitHub.

Si vous ne configurez pas de Node Termination Handler (gestionnaire de terminaison des nœuds), nous vous recommandons de commencer à vider vos Pods manuellement avant d'atteindre le délai de 30 minutes afin qu'ils aient suffisamment de temps pour être vidés de manière gracieuse.

Lancement de nœuds Bottlerocket autogérés

> **Note**
>
> Les groupes de nœuds gérés peuvent offrir certains avantages pour votre cas d'utilisation. Pour plus d'informations, consultez Groupes de nœuds gérés.

Cette rubrique explique comment lancer des groupes Auto Scaling de nœuds Bottlerocket enregistrés auprès de votre cluster Amazon EKS. Bottlerocketest un système d'exploitation open source Linux basé sur AWS lequel vous pouvez exécuter des conteneurs sur des machines virtuelles ou des hôtes bare metal. Une fois que les nœuds ont rejoint le cluster, vous pouvez y déployer des applications Kubernetes. Pour plus d'informations sur Bottlerocket, consultez Utilisation d'une AMI Bottlerocket avec Amazon EKS sur GitHub et Prise en charge des AMI personnalisées dans la documentation de `eksctl`.

Pour plus d'informations sur les mises à niveau sur place, consultez Opérateur de mise à jour Bottlerocket sur GitHub.

> **Important**
>
> - Les nœuds Amazon EKS sont des instances Amazon EC2 standard pour lesquelles vous êtes facturé en fonction du tarif normal pour les instances Amazon EC2. Pour plus d'informations, consultez Tarification Amazon EC2.
> - Vous pouvez lancer des nœuds Bottlerocket dans des clusters étendus Amazon EKS sur AWS Outposts, mais vous ne pouvez pas les lancer dans des clusters locaux sur Outposts. AWS Pour plus d'informations, consultez Amazon EKS sur AWS Outposts.
> - Vous pouvez déployer sur des instances Amazon EC2 avec des processeurs x86 ou Arm. Cependant, vous ne pouvez pas déployer sur des instances qui ont des puces Inferentia.

- Bottlerocketest compatible avec AWS CloudFormation. Cependant, aucun CloudFormation modèle officiel ne peut être copié pour déployer des Bottlerocket nœuds pour Amazon EKS.
- Les images Bottlerocket ne sont pas fournies avec un serveur SSH ou un shell. Vous pouvez utiliser des méthodes out-of-band d'accès pour autoriser l'SSHactivation du conteneur d'administration et pour passer certaines étapes de configuration d'amorçage avec les données utilisateur. Pour plus d'informations, consultez ces sections dans le README.md de bottlerocket sur GitHub :
  - Exploration
  - Conteneur d'administration
  - Paramètres Kubernetes

Pour lancer des nœuds Bottlerocket en utilisant **`eksctl`**

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

> **Note**
>
> Cette procédure fonctionne uniquement pour les clusters créés avec `eksctl`.

1. Copiez les contenus suivants sur votre appareil. Remplacez *`my-cluster`* par le nom de votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Remplacer *`ng-bottlerocket`* avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Pour effectuer un déploiement sur des instances Arm, remplacez *`m5.large`* par un type d'instance Arm. Remplacez *`my-ec2-keypair-name`* par le nom

d'une paire de clés SSH Amazon EC2 que vous pouvez utiliser pour vous connecter à l'aide de SSH dans vos nœuds après leur lancement. Si vous ne possédez pas déjà une paire de clés Amazon EC2, vous pouvez en créer une dans l' AWS Management Console. Pour plus d'informations, veuillez consulter la rubrique Paires de clés Amazon EC2 dans le Guide de l'utilisateur Amazon EC2. Remplacez tous les *example values* restants par vos propres valeurs. Une fois les remplacements effectués, exécutez la commande modifiée pour créer le fichier `bottlerocket.yaml`.

Si vous spécifiez un type d'instance Arm Amazon EC2, consultez les considérations dans AMI Amazon Linux Arm optimisées pour Amazon EKS avant le déploiement. Pour obtenir des instructions sur le déploiement avec une AMI personnalisée, consultez Création de Bottlerocket sur GitHub et Prise en charge des AMI personnalisées dans la documentation `eksctl`. Pour déployer un groupe de nœuds gérés, déployez une AMI personnalisée à l'aide d'un modèle de lancement. Pour plus d'informations, consultez Personnalisation des nœuds gérés avec des modèles de lancement.

> **Important**
>
> Pour déployer un groupe de nœuds ou des sous-réseaux de zone AWS locale, ne transmettez pas AWS Outposts AWS Wavelength, ou des sous-réseaux de zone AWS locale lorsque vous créez le cluster. AWS Outposts AWS Wavelength Vous devez spécifier les sous-réseaux dans l'exemple suivant. Pour plus d'informations, consultez Créer un nodegroup à partir d'un fichier de configuration et Schéma du fichier de configuration dans la documentation `eksctl`. *region-code*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

```
cat >bottlerocket.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.30'

iam:
  withOIDC: true
```

```
nodeGroups:
  - name: ng-bottlerocket
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Bottlerocket
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

2. Deployez vos nœuds avec la commande suivante :

```
eksctl create nodegroup --config-file=bottlerocket.yaml
```

L'exemple qui suit illustre un résultat.

Plusieurs lignes sont affichées pendant la création des nœuds. L'une des dernières lignes de sortie est similaire à la ligne d'exemple suivante.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facultatif) Créez un volume persistant Kubernetes sur un nœud Bottlerocket à l'aide du Plugin CSI Amazon EBS. Le pilote Amazon EBS par défaut se base sur des outils de systèmes de fichiers qui ne sont pas inclus dans Bottlerocket Bottlerocket. Pour obtenir plus d'informations sur la création d'une classe de stockage à l'aide du pilote, consultez Pilote CSI Amazon EBS.
4. (Facultatif) Par défaut, `kube-proxy` définit le paramètre de noyau `nf_conntrack_max` sur une valeur par défaut qui peut différer de ce que Bottlerocket a initialement défini au démarrage. Pour conserver le paramètre par défaut de Bottlerocket, modifiez la configuration `kube-proxy` avec la commande suivante.

```
kubectl edit -n kube-system daemonset kube-proxy
```

Ajoutez `--conntrack-max-per-core` et `--conntrack-min` aux arguments `kube-proxy` présentés dans l'exemple suivant. Un paramètre de `0` signifie aucun changement.

```
      containers:
      - command:
        - kube-proxy
        - --v=2
        - --config=/var/lib/kube-proxy-config/config
        - --conntrack-max-per-core=0
        - --conntrack-min=0
```

5. (Facultatif) Déployez un exemple d'application pour tester vos nœuds Bottlerocket.
6. Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

   - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
   - Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

   Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Lancement de nœuds Windows autogérés

Cette rubrique décrit comment lancer des groupes Auto Scaling de nœuds Windows qui s'inscrivent avec votre cluster Amazon EKS. Une fois que les nœuds ont rejoint le cluster, vous pouvez y déployer des applications Kubernetes.

> **Important**
>
> - Les nœuds Amazon EKS sont des instances Amazon EC2 standard pour lesquelles vous êtes facturé en fonction du tarif normal pour les instances Amazon EC2. Pour plus d'informations, consultez Tarification Amazon EC2.

- Vous pouvez lancer des nœuds Windows dans des clusters étendus Amazon EKS sur AWS Outposts, mais vous ne pouvez pas les lancer dans des clusters locaux sur AWS Outposts. Pour plus d'informations, consultez Amazon EKS sur AWS Outposts.

Activer la prise en charge de Windows pour votre cluster. Nous vous recommandons de passer en revue les considérations importantes avant de lancer un groupe de nœuds Windows. Pour plus d'informations, consultez Activation de la prise en charge de Windows.

Vous pouvez lancer des nœuds Windows autogérés avec `eksctl` ou la AWS Management Console.

eksctl

Pour lancer des nœuds Windows autogérés à l'aide de **`eksctl`**

Cette procédure exige que vous installiez `eksctl` et que votre version `eksctl` soit au moins la version `0.183.0`. Vous pouvez vérifier votre version à l'aide de la commande suivante.

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

> **Note**
>
> Cette procédure fonctionne uniquement pour les clusters créés avec `eksctl`.

1. (Facultatif) Si la politique IAM gérée AmazonEKS_CNI_Policy (si vous avez un cluster `IPv4`) ou la politique *`AmazonEKS_CNI_IPv6_Policy`* (que vous avez créée vous-même si vous avez un cluster `IPv6`) est associée à votre the section called "Rôle IAM de nœud", nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service `aws-node` de Kubernetes à la place. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
2. Cette procédure part du principe que vous disposez d'un cluster existant. Si vous ne disposez pas encore d'un cluster Amazon EKS et d'un groupe de nœuds Amazon Linux auxquels ajouter un groupe de Windows nœuds, nous vous recommandons de suivre le Démarrage

avec Amazon EKS : `eksctl` guide. Ce guide fournit une démonstration complète de la création d'un cluster Amazon EKS avec des nœuds Amazon Linux.

Créez votre groupe de nœuds avec la commande suivante. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Remplacer *`ng-windows`* avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Pour Kubernetes version `1.24` ou ultérieure, vous pouvez remplacer *`2019`* par `2022` pour utiliser Windows Server 2022. Remplacez le reste des *`example values`* par vos propres valeurs.

> **Important**
>
> Pour déployer un groupe de nœuds ou des sous-réseaux de zone AWS locale, ne transmettez pas les sous-réseaux AWS Outposts Wavelength ou Local Zone lorsque vous créez le cluster. AWS Outposts AWS Wavelength Créez le groupe de nœuds avec un fichier de configuration, en spécifiant les AWS Outposts sous-réseaux Wavelength ou Local Zone. Pour plus d'informations, consultez Créer un nodegroup à partir d'un fichier config et Schéma du fichier Config dans la documentation `eksctl`.

```
eksctl create nodegroup \
    --region region-code \
    --cluster my-cluster \
    --name ng-windows \
    --node-type t2.large \
    --nodes 3 \
    --nodes-min 1 \
    --nodes-max 4 \
    --managed=false \
    --node-ami-family WindowsServer2019FullContainer
```

> **Note**
>
> - Si les nœuds ne parviennent pas à rejoindre le cluster, reportez-vous à Les nœuds ne parviennent pas à joindre le cluster dans le guide de dépannage.
> - Pour voir les options disponibles pour les commandes `eksctl`, saisissez la commande suivante.
>
> ```
> eksctl command -help
> ```

L'exemple qui suit illustre un résultat. Plusieurs lignes sont affichées pendant la création des nœuds. L'une des dernières lignes de sortie est similaire à la ligne d'exemple suivante.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facultatif) Déployez un exemple d'application pour tester votre cluster et les nœuds Windows.
4. Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

   - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
   - Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

   Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

AWS Management Console

Prérequis

- Un cluster Amazon EKS existant et un groupe de nœuds Linux. Si vous ne disposez pas de ces ressources, nous vous recommandons de suivre l'un de nos guides Démarrer avec Amazon

EKS pour les créer. Les guides décrivent comment créer un cluster Amazon EKS avec des nœuds Linux.

- Un VPC et un groupe de sécurité existants qui remplissent les conditions requises pour un cluster Amazon EKS. Pour plus d'informations, consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux et Considérations et exigences relatives aux groupes de sécurité Amazon EKS. Le guide Démarrer avec Amazon EKS crée un VPC qui répond aux exigences. Vous pouvez également suivre Création d'un VPC pour votre cluster Amazon EKS pour en créer un manuellement.
- Un cluster Amazon EKS existant qui utilise un VPC et un groupe de sécurité qui remplit les conditions requises pour un cluster Amazon EKS. Pour plus d'informations, consultez Création d'un cluster Amazon EKS. Si vous avez des sous-réseaux Région AWS là où vous l'avez activé AWS Outposts AWS Wavelength, ou si vous avez activé AWS des Zones Locales, ces sous-réseaux ne doivent pas avoir été transmis lorsque vous avez créé le cluster.

Étape 1 : Pour lancer des Windows nœuds autogérés à l'aide du AWS Management Console

1. Attendez que le statut de votre cluster s'affiche soit `ACTIVE`. Si vous lancez vos nœuds avant que le cluster soit actif, les nœuds ne s'enregistrent pas avec le cluster et vous devez les relancer.
2. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation
3. Sélectionnez Créer la pile.
4. Dans Spécifier le modèle, sélectionnez URL Amazon S3.
5. Copiez l'URL suivante et collez-la dans l'URL Amazon S3.

   ```
   https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2023-02-09/amazon-eks-windows-nodegroup.yaml
   ```

6. Sélectionnez Next (Suivant) deux fois.
7. Sur la page Création rapide d'une pile, saisissez les paramètres suivants en conséquence :

   - Nom de la pile : choisissez un nom pour votre pile AWS CloudFormation . Par exemple, vous pouvez l'appeler ***my-cluster*-nodes**.
   - ClusterName: Entrez le nom que vous avez utilisé lors de la création de votre cluster Amazon EKS.

> **Important**
>
> Ce nom doit correspondre exactement au nom que vous avez utilisé dans Étape 1 : créer votre cluster Amazon EKS. Sinon, vos nœuds ne peuvent pas rejoindre le cluster.

- ClusterControlPlaneSecurityGroupe : Choisissez le groupe de sécurité dans la AWS CloudFormation sortie que vous avez générée lors de la création de votre VPC.

  Les étapes suivantes montrent une méthode permettant de récupérer le groupe applicable.

  1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
  2. Choisissez le nom du cluster.
  3. Choisissez l'onglet Networking (Mise en réseau).
  4. Utilisez la valeur Groupes de sécurité supplémentaires comme référence lorsque vous effectuez une sélection dans la liste déroulante des ClusterControlPlaneSecuritygroupes.
- NodeGroupNom : entrez le nom de votre groupe de nœuds. Ce nom peut être utilisé ultérieurement pour identifier le groupe de nœuds Auto Scaling qui est créé pour vos nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.
- NodeAutoScalingGroupMinSize: Entrez le nombre minimum de nœuds que votre groupe Auto Scaling de nœuds peut atteindre.
- NodeAutoScalingGroupDesiredCapacity: Entrez le nombre de nœuds que vous souhaitez atteindre lors de la création de votre pile.
- NodeAutoScalingGroupMaxSize: Entrez le nombre maximum de nœuds que votre groupe Auto Scaling de nœuds peut atteindre.
- NodeInstanceType : Choisissez un type d'instance pour vos nœuds. Pour plus d'informations, consultez Choix d'un type d'instance Amazon EC2.

> **Note**
>
> Les types d'instances pris en charge pour la dernière version du Amazon VPC CNI plugin for pour Kubernetes sont répertoriés dans vpc_ip_resource_limit.go

sur GitHub. Vous devrez peut-être mettre à jour votre version de CNI pour utiliser les derniers types d'instances pris en charge. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

- NodeImageIDSSMParam : prérempli avec le paramètre Amazon EC2 Systems Manager de l'ID d'AMI principal optimisé Amazon EKS actuellement recommandé. Windows Pour utiliser la version complète de Windows, remplacez *Core* par `Full`.
- NodeImageID : (Facultatif) Si vous utilisez votre propre AMI personnalisée (au lieu de l'AMI optimisée pour Amazon EKS), entrez un ID d'AMI de nœud pour votre Région AWS. Si vous spécifiez une valeur pour ce champ, elle remplace toutes les valeurs du champ NodeImageIDSSMParam.
- NodeVolumeTaille : Spécifiez une taille de volume racine pour vos nœuds, en GiB.
- KeyName: Entrez le nom d'une paire de clés SSH Amazon EC2 que vous pourrez utiliser pour vous connecter via SSH à vos nœuds après leur lancement. Si vous ne possédez pas déjà une paire de clés Amazon EC2, vous pouvez en créer une dans l' AWS Management Console. Pour plus d'informations, veuillez consulter la rubrique Paires de clés Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

> **Note**
>
> Si vous ne fournissez pas de paire de clés ici, la AWS CloudFormation pile ne sera pas créée.

- BootstrapArguments: Spécifiez tous les arguments facultatifs à transmettre au script bootstrap du nœud, tels que des `kubelet` arguments supplémentaires en utilisant-`KubeletExtraArgs`.
- DisableIMDSv1 : par défaut, chaque nœud prend en charge le service de métadonnées d'instance version 1 (IMDSv1) et IMDSv2. Vous pouvez désactiver IMDSv1. Pour empêcher les futurs nœuds et Pods du groupe de nœuds d'utiliser MDSv1, définissez DisableIMDSv1 sur true (vrai). Pour de plus amples informations au sujet d'IMDS, consultez Configuration du service des métadonnées d'instance.
- VpcId: Sélectionnez l'ID du VPC que vous avez créé.
- NodeSecurityGroupes : sélectionnez le groupe de sécurité créé pour votre groupe de Linux nœuds lorsque vous avez créé votre VPC. Si vos nœuds Linux sont associés à plusieurs

groupes de sécurité, spécifiez-les tous. C'est le cas, par exemple, si le groupe de nœuds Linux a été créé avec `eksctl`.

- Sous-réseaux : choisissez les sous-réseaux que vous avez créés. Si vous avez créé votre VPC à l'aide des étapes décrites dans Création d'un VPC pour votre cluster Amazon EKS, spécifiez uniquement les sous-réseaux privés du VPC dans lesquels vos nœuds seront lancés.

> **Important**
>
> - Si certains sous-réseaux sont des sous-réseaux publics, leur paramètre d'attribution automatique d'adresse IP publique doit être activé. Si le paramètre n'est pas activé pour le sous-réseau public, les nœuds que vous déployez sur ce sous-réseau public ne se verront pas attribuer d'adresse IP publique et ne pourront pas communiquer avec le cluster ou d'autres AWS services. Si le sous-réseau a été déployé avant le 26 mars 2020 à l'aide de l'un des modèles AWS CloudFormation VPC Amazon EKS ou à `eksctl` l'aide de, l'attribution automatique d'adresses IP publiques est désactivée pour les sous-réseaux publics. Pour plus d'informations sur l'activation de l'attribution d'adresse IP publique pour un sous-réseau, consultez Modification de l'attribut d'adressage `IPv4` public de votre sous-réseau. Si le nœud est déployé sur un sous-réseau privé, il est capable de communiquer avec le cluster et d'autres AWS services via une passerelle NAT.
> - Si les sous-réseaux ne disposent pas d'un accès Internet, assurez-vous de connaître toutes les considérations et les étapes supplémentaires indiqués dans Exigences relatives aux clusters privés.
> - Si vous sélectionnez AWS Outposts des sous-réseaux Wavelength ou Local Zone, les sous-réseaux ne doivent pas avoir été transmis lors de la création du cluster.

8. Confirmez que la pile peut créer des ressources IAM, puis choisissez Create stack (Créer une pile).
9. Lorsque la création de votre pile est terminée, sélectionnez la pile dans la console et choisissez Outputs (Sorties).
10. Enregistrez le NodeInstanceRôle du groupe de nœuds créé. Vous en aurez besoin lors de la configuration de vos nœuds Windows pour Amazon EKS.

Étape 2 : pour autoriser les nœuds à rejoindre votre cluster

1. Vérifiez si vous avez déjà appliqué le `ConfigMap` `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si vous voyez un `ConfigMap` `aws-auth`, mettez-le à jour si nécessaire.

   a. Ouvrez le `ConfigMap` pour le modifier.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Ajoutez de nouvelles entrées `mapRoles` selon vos besoins. Définissez les `rolearn` valeurs selon les valeurs de NodeInstancerôle que vous avez enregistrées dans les procédures précédentes.

```
[...]
data:
  mapRoles: |
- rolearn: <ARN of linux instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: <ARN of windows instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
        - eks:kube-proxy-windows
[...]
```

   c. Enregistrez le fichier et quittez votre éditeur de texte.

3. Si vous avez reçu un message d'erreur indiquant « `Error from server (NotFound): configmaps "aws-auth" not found` », appliquez le stock `ConfigMap`.

   a. Téléchargez la mappe de configuration.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm-windows.yaml
```

b. Dans le `aws-auth-cm-windows.yaml` fichier, définissez les `rolearn` valeurs sur les valeurs de NodeInstancerôle applicables que vous avez enregistrées dans les procédures précédentes. Pour ce faire, utilisez un éditeur de texte ou remplacez *`example values`* et exécutez la commande suivante :

```
sed -i.bak -e 's|<ARN of linux instance role (not instance profile)>|my-node-linux-instance-role|' \
    -e 's|<ARN of windows instance role (not instance profile)>|my-node-windows-instance-role|' aws-auth-cm-windows.yaml
```

> **Important**
>
> - Ne modifiez aucune autre ligne de ce fichier.
> - N'utilisez pas le même rôle IAM pour les nœuds Windows et Linux.

c. Appliquez la configuration. L'exécution de cette commande peut prendre quelques minutes.

```
kubectl apply -f aws-auth-cm-windows.yaml
```

4. Observez le statut de vos nœuds et attendez qu'ils obtiennent le statut `Ready`.

```
kubectl get nodes --watch
```

Saisissez `Ctrl+C` pour revenir à une invite de shell.

> **Note**
>
> Si vous recevez d'autres erreurs concernant les types d'autorisations ou de ressources, consultez Accès non autorisé ou refusé (`kubectl`) dans la rubrique relative à la résolution des problèmes.

Si les nœuds ne parviennent pas à rejoindre le cluster, reportez-vous à Les nœuds ne parviennent pas à joindre le cluster dans le guide de dépannage.

Étape 3 : actions supplémentaires

1. (Facultatif) Déployez un exemple d'application pour tester votre cluster et les nœuds Windows.
2. (Facultatif) Si la politique IAM gérée AmazonEKS_CNI_Policy (si vous avez un cluster `IPv4`) ou la politique *`AmazonEKS_CNI_IPv6_Policy`* (que vous avez créée vous-même si vous avez un cluster `IPv6`) est associée à votre the section called “Rôle IAM de nœud”, nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service `aws-node` de Kubernetes à la place. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
3. Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

   - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
   - Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

   Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Lancement de nœuds Ubuntu autogérés

> **Note**
>
> Les groupes de nœuds gérés peuvent offrir certains avantages pour votre cas d'utilisation. Pour plus d'informations, consultez Groupes de nœuds gérés.

Cette rubrique explique comment lancer des groupes Auto Scaling Ubuntusur des nœuds Amazon Elastic Kubernetes Service (EKS) ou Amazon Elastic Ubuntu ProKubernetes Service (EKS) enregistrés auprès de votre cluster Amazon EKS. Ubuntuet Ubuntu Pro pour EKS sont basés sur le Ubuntu Minimal LTS officiel, incluent le AWS noyau personnalisé développé conjointement avec AWS EKS et spécialement conçu pour EKS. Ubuntu Proajoute une couverture de sécurité

supplémentaire en prenant en charge les périodes de support prolongées d'EKS, le noyaulivepatch, la conformité à la norme FIPS et la capacité d'exécuter un nombre illimité de Pro conteneurs.

Une fois que les nœuds ont rejoint le cluster, vous pouvez y déployer des applications conteneurisées. Pour plus d'informations, consultez la documentation Ubunturelative à la prise en charge des AMI activées AWS et personnalisées dans la `eksctl` documentation.

> **Important**
>
> - Les nœuds Amazon EKS sont des instances Amazon EC2 standard pour lesquelles vous êtes facturé en fonction du tarif normal pour les instances Amazon EC2. Pour plus d'informations, consultez Tarification Amazon EC2.
> - Vous pouvez lancer Ubuntu des nœuds dans des clusters étendus Amazon EKS sur AWS Outposts, mais vous ne pouvez pas les lancer dans des clusters locaux sur AWS Outposts. Pour plus d'informations, consultez Amazon EKS sur AWS Outposts.
> - Vous pouvez déployer sur des instances Amazon EC2 avec des processeurs x86 ou Arm. Toutefois, les instances dotées de Inferentia puces peuvent avoir besoin d'installer d'abord le NeuronSDK.

Pour lancer Ubuntu pour EKS ou Ubuntu Pro pour les nœuds EKS à l'aide de **`eksctl`**

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

> **Note**
>
> Cette procédure fonctionne uniquement pour les clusters créés avec `eksctl`.

1. Copiez les contenus suivants sur votre appareil. Remplacez `my-cluster` par le nom de votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphabétique et ne doit pas dépasser

100 caractères. Remplacer `ng-ubuntu` avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Pour effectuer un déploiement sur Arm des instances, remplacez-le `m5.large` par un type d'Arminstance. Remplacez `my-ec2-keypair-name` par le nom d'une paire de clés SSH Amazon EC2 que vous pouvez utiliser pour vous connecter à l'aide de SSH dans vos nœuds après leur lancement. Si vous ne possédez pas déjà une paire de clés Amazon EC2, vous pouvez en créer une dans l' AWS Management Console. Pour plus d'informations, consultez les paires de clés Amazon EC2 dans le guide de l'utilisateur Amazon EC2. Remplacez tous les *`example values`* restants par vos propres valeurs. Une fois les remplacements effectués, exécutez la commande modifiée pour créer le fichier `ubuntu.yaml`.

> **Important**
>
> Pour déployer un groupe de nœuds ou des sous-réseaux de zone AWS locale, ne transmettez pas AWS Outposts AWS Wavelength, ou des sous-réseaux de zone AWS locale lorsque vous créez le cluster. AWS Outposts AWS Wavelength Vous devez spécifier les sous-réseaux dans l'exemple suivant. Pour plus d'informations, consultez Créer un nodegroup à partir d'un fichier de configuration et Schéma du fichier de configuration dans la documentation `eksctl`. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

```
cat >ubuntu.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.30'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-ubuntu
    instanceType: m5.large
```

```
    desiredCapacity: 3
    amiFamily: Ubuntu22.04
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

Pour créer un groupe de Ubuntu Pro nœuds, il suffit de changer la `amiFamily` valeur en`UbuntuPro2204`.

2. Deployez vos nœuds avec la commande suivante :

```
eksctl create nodegroup --config-file=ubuntu.yaml
```

L'exemple qui suit illustre un résultat.

Plusieurs lignes sont affichées pendant la création des nœuds. L'une des dernières lignes de sortie est similaire à la ligne d'exemple suivante.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facultatif) Déployez un exemple d'application pour tester vos nœuds Ubuntu.
4. Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

   - Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
   - Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

   Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Mises à jour des nœuds autogérés

Quand une nouvelle AMI optimisée pour Amazon EKS est publiée, pensez à remplacer les nœuds autogérés au sein de votre groupe par cette nouvelle AMI. De même, si vous avez mis à jour la version Kubernetes pour votre cluster Amazon EKS, mettez à jour les nœuds afin qu'ils utilisent la même version de Kubernetes.

> **Important**
>
> Cette rubrique couvre les mises à jour des nœuds pour les groupes de nœuds autogérés. Si vous utilisez Groupes de nœuds gérés, consultez Mise à jour d'un groupe de nœuds gérés.

Deux méthodes de base permettent de mettre à jour les groupes de nœuds autogérés dans vos clusters afin d'utiliser une nouvelle AMI :

Migration vers un nouveau groupe de nœud

Créez un nouveau groupe de nœuds et migrez vos Pods vers ce groupe. La migration vers un nouveau groupe de nœuds est plus efficace que la simple mise à jour de l'ID d'AMI dans une pile AWS CloudFormation existante. En effet, le processus de migration rejette l'ancien groupe de nœuds comme `NoSchedule` et purge les nœuds après qu'une nouvelle pile soit prête à accepter la charge de travail des Pod existants.

Mise à jour d'un groupe de nœuds autogérés existant

Mettez à jour la pile AWS CloudFormation d'un groupe de nœuds existant pour utiliser la nouvelle AMI. Cette méthode n'est pas prise en charge pour les groupes de nœuds créés avec `eksctl`.

Migration vers un nouveau groupe de nœud

Cette rubrique vous explique comment créer un groupe de nœuds, migrer avec élégance vos applications existantes vers ce nouveau groupe, puis supprimer l'ancien groupe de nœuds de votre cluster. Vous pouvez migrer vers un nouveau groupe de nœuds à l'aide de`eksctl` ou de AWS Management Console.

eksctl

Pour migrer vos applications vers un nouveau groupe de nœuds avec **`eksctl`**

Pour plus d'informations sur l'utilisation d'eksctl pour la migration, consultez la section Groupes de nœuds non gérés dans la documentation. `eksctl`

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

> **Note**
>
> Cette procédure fonctionne uniquement pour les clusters et les groupes de nœuds créés avec `eksctl`.

1. Récupérez le nom de vos groupes de nœuds existants, en remplaçant *`my-cluster`* par le nom de votre cluster.

   ```
   eksctl get nodegroups --cluster=my-cluster
   ```

   L'exemple qui suit illustre un résultat.

   ```
   CLUSTER      NODEGROUP          CREATED               MIN SIZE      MAX SIZE
    DESIRED CAPACITY     INSTANCE TYPE     IMAGE ID
   default      standard-nodes   2019-05-01T22:26:58Z  1             4             3
                       t3.medium         ami-05a71d034119ffc12
   ```

2. Lancez un nouveau groupe de nœuds avec `eksctl` à l'aide de la commande suivante. Dans la commande, remplacez chaque *`example value`* par vos propres valeurs. Le numéro de version ne peut pas être postérieur à la version Kubernetes de votre plan de contrôle. Il ne peut pas non plus être antérieur de plus de deux versions mineures à la version Kubernetes de votre plan de contrôle. Nous vous recommandons d'utiliser la même version que votre plan de contrôle.

Nous recommandons de bloquer l'accès des Pod à IMDS si les conditions suivantes sont remplies :

- Vous envisagez d'attribuer des rôles IAM à tous vos comptes de service Kubernetes afin que les Pods ne disposent que des autorisations minimales dont ils ont besoin.
- Aucun Pods membre du cluster n'a besoin d'accéder au service de métadonnées d'instance Amazon EC2 (IMDS) pour d'autres raisons, telles que la récupération du courant. Région AWS

Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Pour bloquer l'accès des Pod à IMDS, ajoutez l'option `--disable-pod-imds` à la commande suivante.

> **Note**
>
> Pour plus d'indicateurs disponibles et leurs descriptions, consultez https://eksctl.io/.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --version 1.30 \
  --name standard-nodes-new \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --managed=false
```

3. Lorsque la commande précédente se termine, vérifiez que tous vos nœuds ont atteint l'état `Ready` à l'aide de la commande suivante :

```
kubectl get nodes
```

4. Supprimez le groupe de nœuds d'origine avec la commande suivante. Dans la commande, remplacez chaque *example value* par les noms de votre cluster et de votre groupe de nœuds :

```
eksctl delete nodegroup --cluster my-cluster --name standard-nodes-old
```

AWS Management Console and AWS CLI

Pour migrer vos applications vers un nouveau groupe de nœuds à l'aide AWS Management Console des AWS CLI

1. Lancez un nouveau groupe de nœuds en suivant les étapes décrites dans Lancement de nœuds Amazon Linux autogérés.
2. Lorsque la création de votre pile est terminée, sélectionnez la pile dans la console et choisissez Outputs (Sorties).
3. Enregistrez le NodeInstancerôle du groupe de nœuds créé. Vous en aurez besoin pour ajouter les nouveaux nœuds Amazon EKS à votre cluster.

> **Note**
>
> Si vous avez attachés des politiques IAM supplémentaires au rôle IAM de votre ancien groupe de nœuds, attachez ces mêmes politiques au rôle IAM de votre nouveau groupe de nœuds pour maintenir cette fonctionnalité sur le nouveau groupe. Cela s'applique si vous avez ajouté des autorisations pour le Kubernetes Cluster Autoscaler, par exemple.

4. Mettez à jour les groupes de sécurité pour les deux groupes de nœuds afin qu'ils puissent communiquer entre eux. Pour plus d'informations, consultez Considérations et exigences relatives aux groupes de sécurité Amazon EKS.

   a. Notez l'ID des groupes de sécurité des deux groupes de nœuds. Ceci est affiché sous forme de valeur de NodeSecuritygroupe dans les sorties de la AWS CloudFormation pile.

      Vous pouvez utiliser les AWS CLI commandes suivantes pour obtenir les identifiants des groupes de sécurité à partir des noms des piles. Dans ces commandes, `oldNodes` il s'agit du nom de la AWS CloudFormation pile de votre ancienne pile de nœuds et `newNodes` du nom de la pile vers laquelle vous effectuez la migration. Remplacez chaque *`example value`* par vos propres valeurs.

      ```
      oldNodes="old_node_CFN_stack_name"
      newNodes="new_node_CFN_stack_name"
      ```

```
oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $oldNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $newNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
```

b. Ajoutez des règles d'entrée pour chaque groupe de sécurité des nœuds afin d'accepter le trafic de l'autre.

   Les AWS CLI commandes suivantes ajoutent des règles entrantes à chaque groupe de sécurité qui autorisent tout le trafic sur tous les protocoles en provenance de l'autre groupe de sécurité. Cette configuration permet aux Pods au sein de chaque groupe de nœuds de communiquer entre eux pendant que vous migrez la charge de travail vers le nouveau groupe.

```
aws ec2 authorize-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 authorize-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

5. Modifiez la configmap `aws-auth` afin de mapper le nouveau rôle d'instance des nœuds dans RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Ajoutez une nouvelle entrée `mapRoles` pour le nouveau groupe de nœuds. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: ARN of instance role (not instance profile)
      username: system:node:{{EC2PrivateDNSName}}
      groups:
```

```
        - system:bootstrappers
        - system:nodes>
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

Remplacez l'*ARN of instance role (not instance profile)*extrait par la valeur du NodeInstancerôle que vous avez enregistrée lors d'une étape précédente. Ensuite, enregistrez et fermez le fichier pour appliquer le configmap mise à jour.

6. Vérifiez le statut de vos nœuds et attendez que vos nouveaux nœuds rejoignent votre cluster et affichent le statut `Ready`.

```
kubectl get nodes --watch
```

7. (Facultatif) Si vous utilisez le composant Kubernetes Cluster Autoscaler, diminuez le déploiement à zéro (0) réplica pour éviter les actions de mise à l'échelle contradictoires.

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

8. Utilisez la commande suivante pour rejeter chacun des nœuds que vous souhaitez supprimer avec `NoSchedule`. Ceci afin que les nouveaux Pods ne soient pas programmés ou reprogrammés sur les nœuds que vous remplacez. Pour plus d'informations, consultez Rejets et les tolérances (français non garanti) dans la documentation de Kubernetes.

```
kubectl taint nodes node_name key=value:NoSchedule
```

Si vous mettez à niveau vos nœuds vers une nouvelle version de Kubernetes, vous pouvez identifier et rejeter tous les nœuds d'une version Kubernetes donnée (dans ce cas `1.28`) avec le snippet de code suivant. Le numéro de version ne peut pas être postérieur à la version Kubernetes de votre plan de contrôle. Il ne peut pas non plus être antérieur de plus de deux versions mineures à la version Kubernetes de votre plan de contrôle. Nous vous recommandons d'utiliser la même version que votre plan de contrôle.

```
K8S_VERSION=1.28
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
```

```
for node in ${nodes[@]}
do
    echo "Tainting $node"
    kubectl taint nodes $node key=value:NoSchedule
done
```

9. Déterminez le fournisseur DNS de votre cluster

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

L'exemple qui suit illustre un résultat. Ce cluster utilise CoreDNS pour la résolution DNS, mais votre cluster peut renvoyer la valeur `kube-dns` à la place) :

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

10. Si votre déploiement actuel exécute moins de deux réplicas, dimensionnez le déploiement à deux réplicas. Remplacez *coredns* par **kubedns** si la sortie de votre commande a plutôt renvoyé cette valeur.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

11. Drainez chacun des nœuds que vous souhaitez supprimer de votre cluster à l'aide de la commande suivante :

```
kubectl drain node_name --ignore-daemonsets --delete-local-data
```

Si vous mettez à niveau vos nœuds vers une nouvelle version de Kubernetes, identifiez et purgez tous les nœuds d'une versionKubernetes donnée (dans ce cas *1.28*) avec le snippet de code suivant.

```
K8S_VERSION=1.28
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Draining $node"
    kubectl drain $node --ignore-daemonsets --delete-local-data
done
```

12. Une fois que vos anciens nœuds ont été purgés, révoquez les règles d'entrée du groupe de sécurité que vous avez autorisées précédemment. Supprimez ensuite la AWS CloudFormation pile pour mettre fin aux instances.

> **Note**
>
> Si vous avez associé des politiques IAM supplémentaires à votre ancien groupe de nœuds (rôle IAM, par exemple en ajoutant des autorisations pour le Kubernetes Cluster Autoscaler), détachez-les du rôle avant de supprimer votre pile. AWS CloudFormation

a. Révoquez les règles entrantes que vous aviez créées précédemment pour les groupes de sécurité de vos nœuds. Dans ces commandes, `oldNodes` il s'agit du nom de la AWS CloudFormation pile de votre ancienne pile de nœuds et `newNodes` du nom de la pile vers laquelle vous effectuez la migration.

```
oldNodes="old_node_CFN_stack_name"
newNodes="new_node_CFN_stack_name"

oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $oldNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $newNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
aws ec2 revoke-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 revoke-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

b. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

c. Sélectionnez votre ancienne pile de nœuds.

d. Sélectionnez Delete (Supprimer).

e. Dans la boîte de dialogue de confirmation Delete stack (Supprimer la pile), choisissez Delete stack (Supprimer la pile).

13. Modifiez la configmap `aws-auth` afin de supprimer l'ancien rôle d'instance des nœuds dans RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Supprimez l'entrée `mapRoles` de l'ancien groupe de nœuds. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
W70725MZQFF8
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-15-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
```

Enregistrez et fermez le fichier afin d'appliquer la configmap mise à jour.

14. (Facultatif) Si vous utilisez le composant Kubernetes Cluster Autoscaler, dimensionnez le déploiement à un réplica.

> **Note**
>
> Vous devez également labeliser votre nouveau groupe Auto Scaling de façon appropriée (par exemple `k8s.io/cluster-autoscaler/enabled`,`k8s.io/cluster-autoscaler/`*`my-cluster`*) et mettre à jour la commande de votre déploiement Cluster Autoscaler afin qu'elle pointe vers le groupe Auto Scaling

nouvellement labelisé. Pour plus d'informations, consultez Cluster Autoscaler activé. AWS

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

15. (Facultatif) Vérifiez que vous utilisez la dernière version du plugin CNI Amazon VPC pour Kubernetes. Vous devrez peut-être mettre à jour votre version de CNI pour utiliser les derniers types d'instance pris en charge. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.
16. Si votre cluster utilise `kube-dns` pour la résolution DNS (voir l'étape précédente), dimensionnez le déploiement `kube-dns` à un réplica.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

Mise à jour d'un groupe de nœuds autogérés existant

Cette rubrique décrit comment mettre à jour une pile de nœuds AWS CloudFormation autogérés existante avec une nouvelle AMI. Vous pouvez utiliser cette procédure pour mettre à jour vos nœuds vers une nouvelle version de Kubernetes suite à une mise à jour du cluster. Sinon, vous pouvez effectuer une mise à jour vers la dernière AMI optimisée pour Amazon EKS pour une version Kubernetes existante.

Important

Cette rubrique couvre les mises à jour des nœuds pour les groupes de nœuds autogérés. Pour obtenir des informations sur l'utilisation d'Groupes de nœuds gérés, veuillez consulter Mise à jour d'un groupe de nœuds gérés.

Le dernier AWS CloudFormation modèle de nœud Amazon EKS par défaut est configuré pour lancer une instance avec la nouvelle AMI dans votre cluster avant de supprimer l'ancienne, une par une. Cette configuration permet de garantir que vous disposez toujours du nombre d'instances actives souhaité de votre groupe Auto Scaling dans votre cluster lors de la propagation des mises à jour.

> **Note**
>
> Cette méthode n'est pas prise en charge pour les groupes de nœud créés avec `eksctl`. Si vous avez créé votre cluster ou votre groupe de nœuds avec `eksctl`, consultez Migration vers un nouveau groupe de nœud.

Pour mettre à jour un groupe de nœuds existant

1. Déterminez le fournisseur DNS de votre cluster.

   ```
   kubectl get deployments -l k8s-app=kube-dns -n kube-system
   ```

   L'exemple qui suit illustre un résultat. Ce cluster utilise CoreDNS pour la résolution DNS, mais votre cluster peut renvoyer la valeur `kube-dns` à la place. Votre sortie peut être différente selon la version de `kubectl` que vous utilisez.

   ```
   NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
   coredns   1         1         1            1           31m
   ```

2. Si votre déploiement actuel exécute moins de deux réplicas, dimensionnez le déploiement à deux réplicas. Remplacez *coredns* par **kube-dns** si la sortie de votre commande a plutôt renvoyé cette valeur.

   ```
   kubectl scale deployments/coredns --replicas=2 -n kube-system
   ```

3. (Facultatif) Si vous utilisez le composant Cluster Autoscaler de Kubernetes, diminuez le déploiement à zéro (0) réplica pour éviter les actions de mise à l'échelle contradictoires.

   ```
   kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
   ```

4. Déterminez le type d'instance et le nombre d'instances souhaité pour votre groupe actuel de nœuds. Vous saisirez ces valeurs ultérieurement lorsque vous mettrez à jour le AWS CloudFormation modèle du groupe.

   a. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.

   b. Dans le panneau de navigation de gauche, sélectionnez Launch Configurations (Configurations du lancement) et notez le type d'instance pour votre configuration actuelle de lancement des nœuds.

c. Dans le panneau de navigation de gauche, sélectionnez Auto Scaling Groups (Groupes Auto Scaling) et notez le nombre d'instances souhaité pour le groupe Auto Scaling actuel des nœuds.

5. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
6. Sélectionnez votre pile de groupe de nœuds, puis choisissez Update (Mettre à jour).
7. Sélectionnez Replace current template (Remplacer le modèle actuel) et sélectionnez Amazon S3 URL (URL Amazon S3).
8. Pour l'URL Amazon S3, collez l'URL suivante dans la zone de texte pour vous assurer que vous utilisez la dernière version du AWS CloudFormation modèle de nœud. Ensuite, sélectionnez Next (Suivant) :

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
```

9. Sur la page Specify stack details (Spécifier les détails de la pile), renseignez les paramètres suivants, puis choisissez Next (Suivant) :

   - NodeAutoScalingGroupDesiredCapacity— Entrez le nombre d'instances souhaité que vous avez enregistré lors d'une étape précédente. Ou saisissez le nouveau nombre de nœuds souhaité pour la mise à l'échelle lorsque votre pile sera mise à jour.
   - NodeAutoScalingGroupMaxSize— Entrez le nombre maximum de nœuds auxquels votre groupe Auto Scaling de nœuds peut s'étendre. Cette valeur doit être supérieure d'au moins un nœud à votre capacité souhaitée. Ceci afin de pouvoir effectuer une mise à jour continue de vos nœuds sans réduire votre nombre de nœuds pendant la mise à jour.
   - NodeInstanceType — Choisissez le type d'instance que vous avez enregistré à l'étape précédente. Vous pouvez également choisir un type d'instance différent pour vos nœuds. Avant de choisir un type d'instance, consultez Choix d'un type d'instance Amazon EC2. Chaque type d'instance Amazon EC2 prend en charge un nombre maximal d'interfaces réseau Elastic (interface réseau) et chaque interface réseau prend en charge un nombre maximal d'adresses IP. Étant donné que chaque composant master et Pod se voit attribuer sa propre adresse IP, il est important de choisir un type d'instance qui prendra en charge le nombre maximal de Pods que vous souhaitez exécuter sur chaque nœud Amazon EC2. Pour une liste du nombre d'interfaces réseau et d'adresses IP pris en charge par les types d'instances, consultez adresses IP par interface réseau et par type d'instance. Par exemple, le type

d'instance `m5.large` prend en charge un maximum de 30 adresses IP pour le composant master et les Pods.

> **Note**
>
> Les types d'instances pris en charge pour la dernière version du Amazon VPC CNI plugin for pour Kubernetes sont affichés dans vpc_ip_resource_limit.go sur GitHub. Vous devrez peut-être mettre à jour votre version Amazon VPC CNI plugin for Kubernetes pour utiliser les derniers types d'instances pris en charge. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

> **Important**
>
> Certains types d'instances peuvent ne pas être disponibles dans tous les cas Régions AWS.

- NodeImageIDSSMParam — Le paramètre Amazon EC2 Systems Manager de l'ID AMI vers lequel vous souhaitez effectuer la mise à jour. La valeur suivante utilise la dernière AMI optimisée pour Amazon EKS pour Kubernetes version `1.30`.

```
/aws/service/eks/optimized-ami/1.30/amazon-linux-2/recommended/image_id
```

Vous pouvez remplacer la version *1.30* par une version de Kubernetes prise en charge qui est la même. Ou bien, elle doit être antérieure d'une version au maximum à la version Kubernetes exécutée sur votre plan de contrôle. Nous vous recommandons de conserver vos nœuds dans la même version que votre plan de contrôle. Vous pouvez également le remplacer *amazon-linux-2* par un autre type d'AMI. Pour plus d'informations, consultez Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS.

> **Note**
>
> L'utilisation du paramètre Amazon EC2 Systems Manager vous permet de mettre à jour vos nœuds à l'avenir sans avoir à rechercher et spécifier un ID d'AMI. Si votre pile AWS CloudFormation utilise cette valeur, toute mise à jour de pile lance toujours la dernière AMI optimisée pour Amazon EKS recommandée pour votre version

Kubernetes spécifiée. C'est le cas même si vous ne modifiez aucune valeur dans le modèle.

- NodeImageID — Pour utiliser votre propre AMI personnalisée, entrez l'ID de l'AMI à utiliser.

> **Important**
>
> Cette valeur remplace toute valeur spécifiée pour NodeImageIDSSMParam. Si vous souhaitez utiliser la valeur NodeImageIDSSMParam, assurez-vous que la valeur de Id est vide. NodeImage

- DisableIMDSv1 : par défaut, chaque nœud prend en charge le service de métadonnées d'instance version 1 (IMDSv1) et IMDSv2. Cependant, vous pouvez désactiver IMDSv1. Sélectionnez vrai si vous ne souhaitez pas qu'un nœud ou des Pods programmés dans le groupe de nœuds utilisent IMDSv1. Pour de plus amples informations au sujet d'IMDS, consultez Configuration du service des métadonnées d'instance. Si vous avez implémenté des rôles IAM pour les comptes de service, attribuez les autorisations nécessaires directement à tous ceux Pods qui ont besoin d'accéder aux AWS services. Ainsi, aucun Pods membre de votre cluster n'a besoin d'accéder à l'IMDS pour d'autres raisons, telles que la récupération du courant. Région AWS Ensuite, vous pouvez également désactiver l'accès à IMDSv2 pour les Pods qui n'utilisent pas de réseau hôte. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

10. (Facultatif) Sur la page Options (Options), étiquetez les ressources de votre pile. Choisissez Next (Suivant).
11. Sur la page Review (Vérification), vérifiez vos informations, acceptez que la pile puisse créer des ressources IAM, puis choisissez Update stack (Mettre à jour la pile).

> **Note**
>
> La mise à jour de chaque nœud du cluster prend plusieurs minutes. Attendez que la mise à jour de tous les nœuds soit terminée avant d'effectuer les étapes suivantes.

12. Si le fournisseur DNS de votre cluster est `kube-dns`, dimensionnez le déploiement `kube-dns` à un réplica.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

13. (Facultatif) Si vous utilisez le composant Cluster Autoscaler de Kubernetes, redimensionnez le déploiement à la quantité souhaitée de réplicas.

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

14. (Facultatif) Vérifiez que vous utilisez la dernière version du Amazon VPC CNI plugin for Kubernetes. Vous devrez peut-être mettre à jour votre version Amazon VPC CNI plugin for Kubernetes pour utiliser les derniers types d'instances pris en charge. Pour plus d'informations, voir Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

AWS Fargate

> **Important**
>
> AWS Fargate avec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Cette rubrique porte sur l'utilisation d'Amazon EKS pour exécuter les Pods Kubernetes sur AWS Fargate. Fargate est une technologie qui fournit une capacité de calcul à la demande et de taille appropriée pour les conteneurs. Avec Fargate, il n'est pas nécessaire d'allouer, de configurer ou de mettre à l'échelle des groupes de machines virtuelles pour exécuter les conteneurs. Vous n'avez plus à choisir les types de serveurs, à décider quand vos clusters de nœuds doivent être mis à l'échelle, ni à optimiser les packs de clusters.

Vous pouvez contrôler les Pods qui démarrent sur Fargate et la manière dont ils s'exécutent avec des profils Fargate. Les profils Fargate sont définis dans le cadre de votre cluster Amazon EKS. Amazon EKS s'Kubernetesintègre à Fargate en utilisant des contrôleurs conçus à l'aide du modèle extensible en amont fourni AWS par. Kubernetes Ces contrôleurs font partie du plan de contrôle Kubernetes géré par Amazon EKS et sont responsables de la programmation des Pods Kubernetes natifs sur Fargate. Les contrôleurs Fargate comprennent un nouveau planificateur qui s'exécute parallèlement au planificateur Kubernetes par défaut, outre plusieurs contrôleurs d'admission mutants et validants. Lorsque vous démarrez un Pod qui répond aux critères d'exécution sur Fargate, les contrôleurs Fargate qui s'exécutent dans le cluster reconnaissent, mettent à jour et programment le Pod sur Fargate.

Cette rubrique décrit les différents composants des Pods qui s'exécutent sur Fargate, et appelle à des considérations particulières pour l'utilisation de Fargate avec Amazon EKS.

AWS Fargate considérations

Voici quelques éléments à prendre en compte pour l'utilisation de Fargate sur Amazon EKS.

- Chaque Pod qui s'exécute sur Fargate a sa propre limite d'isolement. Ils ne partagent pas le noyau sous-jacent, les ressources du processeur, les ressources mémoire ou l'interface réseau Elastic avec un autre Pod.
- Les Network Load Balancers et les Application Load Balancers (ALB) peuvent être utilisés avec Fargate avec des cibles IP uniquement. Pour plus d'informations, consultez Créer un équilibreur de charge de réseau et Répartition de la charge des applications sur Amazon EKS.
- Les services exposés de Fargate s'exécutent uniquement en mode IP de type cible, et non en mode IP de nœud. La manière recommandée de vérifier la connectivité entre un service s'exécutant sur un nœud géré et un service s'exécutant sur Fargate est de se connecter via le nom du service.
- Les pods doivent correspondre à un profil Fargate au moment où ils sont programmés pour fonctionner sur Fargate. Les pods qui ne correspondent pas à un profil Fargate peuvent être bloqués comme `Pending`. Si un profil Fargate correspondant existe, vous pouvez supprimer les Pods en attente que vous avez créés pour les reprogrammer sur Fargate.
- Les DaemonSets ne sont pas pris en charge sur Fargate. Si votre application nécessite un démon, reconfigurez ce démon pour qu'il s'exécute en tant que conteneur secondaire dans vos Pods.
- Les conteneurs privilégiés ne sont pas pris en charge sur Fargate.
- Les pods qui s'exécutent sur Fargate ne peuvent pas spécifier `HostPort` ou `HostNetwork` dans le manifeste du Pod.
- La limite flexible `nofile` et `nproc` par défaut est de 1 024 et la limite stricte est de 65 535 pour les Pods Fargate.
- Les GPU ne sont actuellement pas disponibles sur Fargate.
- Les pods qui s'exécutent sur Fargate ne sont pris en charge que sur les sous-réseaux privés (avec accès par passerelle NAT AWS aux services, mais pas de route directe vers une passerelle Internet). Le VPC de votre cluster doit donc disposer de sous-réseaux privés disponibles. Pour les clusters sans accès Internet sortant, veuillez consulter Exigences relatives aux clusters privés.
- Vous pouvez utiliser l'Vertical Pod Autoscaler (VPA) pour paramétrer la bonne taille initiale du processeur et de la mémoire de vos Pods Fargate, puis utiliser l'Horizontal Pod Autoscaler

pour mettre à l'échelle ces Pods. Si vous souhaitez que le Vertical Pod Autoscaler redéploie automatiquement les Pods vers Fargate avec des combinaisons de processeur et de mémoire plus importantes, définissez le mode de Vertical Pod Autoscale sur `Auto` ou `Recreate` pour garantir une fonctionnalité correcte. Pour plus d'informations, consultez la documentation Vertical Pod Autoscaler sur GitHub.

- La résolution DNS et les noms d'hôte DNS doivent être activés pour votre VPC. Pour plus d'informations, consultez Affichage et mise à jour de la prise en charge de DNS pour votre VPC.
- Amazon EKS Fargate defense-in-depth ajoute Kubernetes des applications en isolant chaque pod au sein d'une machine virtuelle (VM). Cette frontière VM empêche l'accès aux ressources basées sur l'hôte utilisées par d'autres pods en cas de fuite du conteneur, qui est une méthode courante d'attaque des applications conteneurisées et d'accès aux ressources en dehors du conteneur.

  L'utilisation d'Amazon EKS ne modifie pas vos responsabilités dans le cadre du modèle de responsabilité partagée. Vous devez examiner attentivement la configuration des contrôles de sécurité et de gouvernance du cluster. Le moyen le plus sûr d'isoler une application est toujours de l'exécuter dans un cluster séparé.
- Les profils Fargate prennent en charge la spécification de sous-réseaux à partir de blocs CIDR secondaires de VPC. Vous pouvez spécifier un bloc CIDR secondaire. En effet, le nombre d'adresses IP disponibles dans un sous-réseau est limité. Par conséquent, il existe également un nombre limité de Pods qui peuvent être créés dans le cluster. En utilisant différents sous-réseaux pour les Pods, vous pouvez augmenter le nombre d'adresses IP disponibles. Pour plus d'informations, consultez Ajout de blocs CIDR `IPv4` à un VPC.
- Le service de métadonnées d'instance Amazon EC2 (IMDS) n'est pas disponible pour les Pods qui sont déployés sur des nœuds Fargate. Si vous disposez de Pods déployés sur Fargate qui ont besoin d'informations d'identification IAM, attribuez-les à vos Pods en utilisant Rôles IAM pour les comptes de service. Si vos Pods ont besoin d'accéder à d'autres informations disponibles via IMDS, vous devez coder ces informations en dur dans la spécification de votre Pod. Cela inclut la Région AWS ou la zone de disponibilité dans laquelle a Pod est déployé.
- Vous ne pouvez pas déployer Pods Fargate dans des zones locales ou dans AWS Outposts des AWS Wavelength zones locales AWS .
- Amazon EKS doit régulièrement appliquer des correctifs aux Pods Fargate pour assurer leur sécurité. Les tentatives de mise à jour sont réalisées de manière à réduire les répercussions éventuelles, toutefois, il est parfois nécessaire de supprimer des Pods si leur expulsion a échoué. Certaines actions peuvent être prises pour minimiser les perturbations. Pour de plus amples informations, veuillez consulter Application de correctifs au système d'exploitation Fargate.

- Le pluginAmazon VPC CNI pour Amazon EKS est installé sur des nœuds Fargate. Vous ne pouvez pas utiliser Autres plugins CNI compatibles avec des nœuds Fargate.
- Un Pod exécuté sur Fargate monte automatiquement un système de fichiers Amazon EFS. Vous ne pouvez pas utiliser l'approvisionnement dynamique des volumes persistants avec les nœuds Fargate, mais vous pouvez utiliser l'approvisionnement statique.
- Il n'est pas possible de monter des volumes Amazon EBS sur les Pods Fargate.
- Vous pouvez exécuter le contrôleur CSI Amazon EBS sur les nœuds Fargate, mais le nœud CSI DaemonSet Amazon EBS ne peut fonctionner que sur des instances Amazon EC2.
- Une fois qu'une Job Kubernetes est marquée `Completed` ou `Failed`, les Pods que la Job crée continuent normalement d'exister. Ce comportement vous permet de consulter vos journaux et vos résultats, mais avec Fargate, vous devrez payer des frais si vous ne nettoyez pas la Job par la suite.

  Pour supprimer automatiquement les informations associées Pods après la Job fin ou l'échec d'une opération, vous pouvez spécifier une période à l'aide du contrôleur time-to-live (TTL). L'exemple suivant montre comment spécifier `.spec.ttlSecondsAfterFinished` dans le manifeste de votre Job.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: busybox
spec:
  template:
    spec:
      containers:
      - name: busybox
        image: busybox
        command: ["/bin/sh", "-c", "sleep 10"]
      restartPolicy: Never
  ttlSecondsAfterFinished: 60 # <-- TTL controller
```

Commencer à AWS Fargate utiliser Amazon EKS

> **Important**
>
> AWS Fargate avec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Cette rubrique décrit comment commencer à exécuter Pods votre cluster Amazon EKS. AWS Fargate

Si vous limitez l'accès au point de terminaison public de votre cluster à l'aide de blocs CIDR, nous vous recommandons d'activer également l'accès au point de terminaison privé. De cette manière, les Pods Fargate peuvent communiquer avec le cluster. Sans l'activation du point de terminaison privé, les blocs CIDR que vous spécifiez pour l'accès public doivent inclure les sources de sortie de votre VPC. Pour plus d'informations, consultez Contrôle d'accès au point de terminaison du cluster Amazon EKS..

Prérequis

Un cluster existant. Si vous ne disposez pas déjà d'un cluster Amazon EKS, consultez Démarrer avec Amazon EKS.

S'assurer que les nœuds existants peuvent communiquer avec les Pods Fargate

Si vous utilisez un nouveau cluster sans nœuds ou un cluster avec uniquement des groupes de nœuds gérés, vous pouvez passer à Création d'un rôle d'exécution de Pod Fargate.

Supposons que vous travaillez avec un cluster existant ayant déjà des nœuds associés. Assurez-vous que les Pods sur ces nœuds peuvent communiquer librement avec les Pods qui s'exécutent sur Fargate. Les Pods qui s'exécutent sur Fargate sont automatiquement configurés pour utiliser le groupe de sécurité du cluster auquel ils sont associés. Assurez-vous que tous les nœuds existants de votre cluster peuvent envoyer et recevoir du trafic vers et depuis le groupe de sécurité du cluster. Les Groupes de nœuds gérés sont automatiquement configurés pour utiliser également le groupe de sécurité du cluster, vous n'avez donc pas besoin de les modifier ou de les vérifier pour cette compatibilité.

Pour les groupes de nœuds existants créés avec `eksctl` ou avec les AWS CloudFormation modèles gérés par Amazon EKS, vous pouvez ajouter le groupe de sécurité du cluster aux nœuds manuellement. Vous pouvez également modifier le modèle de lancement du groupe Auto Scaling

pour le groupe de nœuds, afin d'attacher le groupe de sécurité du cluster aux instances. Pour plus d'informations, consultez Modification des groupes de sécurité d'une instance dans le Guide de l'utilisateur VPC Amazon.

Vous pouvez rechercher un groupe de sécurité pour votre cluster dans la AWS Management Console section Mise en réseau du cluster. Vous pouvez également le faire à l'aide de la AWS CLI commande suivante. Lorsque vous utilisez cette commande, remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Création d'un rôle d'exécution de Pod Fargate

Lorsque votre cluster est créé Pods AWS Fargate, les composants qui s'exécutent sur l'infrastructure Fargate doivent appeler les API en votre AWS nom. Pour ce faire, le rôle d'exécution de Pod Amazon EKS fournit les autorisations IAM. Pour créer un rôle AWS Fargate Pod d'exécution, voirRôle IAM d'exécution de Pod Amazon EKS.

> **Note**
>
> Si vous avez créé votre cluster avec `eksctl` à l'aide de l'option `--fargate`, votre cluster comporte déjà un rôle d'exécution de Pod que vous pouvez trouver dans la console IAM en utilisant le modèle `eksctl-`*`my-cluster`*`-FargatePodExecutionRole-`*`ABCDEFGHIJKL`*. De même, si vous utilisez `eksctl` pour créer vos profils Fargate, `eksctl` créera votre rôle d'exécution de Pod si celui-ci n'existe pas déjà.

Création d'un profil Fargate pour votre cluster

Pour programmer des Pods qui peuvent s'exécuter sur Fargate dans votre cluster, vous devez définir un profil Fargate qui spécifie quels Pods utilisent Fargate lorsqu'ils sont lancés. Pour plus d'informations, consultez AWS Fargate profil.

> **Note**
>
> Si vous avez créé votre cluster avec `eksctl` à l'aide de l'option `--fargate`, un profil Fargate est déjà créé pour votre cluster avec des sélecteurs pour tous les Pods dans les

espaces de noms `kube-system` et `default`. Utilisez la procédure suivante pour créer des profils Fargate pour tout autre espace de noms que vous souhaitez utiliser avec Fargate.

Vous pouvez créer un profil Fargate en utilisant `eksctl` ou la AWS Management Console.

eksctl

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

Pour créer un profil Fargate avec **`eksctl`**

Créez votre profil Fargate avec la commande `eksctl` suivante, en remplaçant chaque *`example value`* par vos propres valeurs. Vous devez spécifier un espace de noms. Cependant, l'option `--labels` n'est pas obligatoire.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

Vous pouvez utiliser certains caractères génériques pour les étiquettes *`my-kubernetes-namespace`* et *`key=value`*. Pour plus d'informations, consultez Caractères génériques de profils Fargate.

AWS Management Console

Pour créer un profil Fargate pour un cluster avec le AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le cluster pour lequel vous voulez créer un profil Fargate.

3. Choisissez l'onglet Calcul.
4. Sous Fargate profiles (Profils Fargate), choisissez Add Fargate profile (Ajouter un profil Fargate).
5. Sur la page Configure Fargate profile (Configurer le profil Fargate), procédez comme suit :

   a. Dans Nom, saisissez un nom pour votre profil Fargate. Le nom doit être unique.

   b. Pour Rôle d'exécution du pod, choisissez le rôle d'exécution du Pod à utiliser avec votre profil Fargate. Seuls les rôles IAM avec le principal de service `eks-fargate-pods.amazonaws.com` sont affichés. Si vous ne voyez aucun rôle répertorié ici, vous devez en créer un. Pour plus d'informations, consultez Rôle IAM d'exécution de Pod Amazon EKS.

   c. Modifiez les sous-réseaux sélectionnés selon vos besoins.

      > **Note**
      >
      > Seuls les sous-réseaux privés sont pris en charge pour les Pods qui s'exécutent sur Fargate.

   d. Dans Identifications, vous pouvez éventuellement étiqueter votre profil Fargate. Ces balises ne se propagent pas aux autres ressources qui sont associées au profil, comme les Pods.

   e. Choisissez Suivant.
6. Sur la page Configurer la sélection de Pod, procédez comme suit :

   a. Pour Espace de noms, saisissez un espace de noms correspondant aux Pods.

      - Vous pouvez utiliser des espaces de noms spécifiques pour les faire correspondre, tels que **`kube-system`** ou **`default`**.
      - Vous pouvez utiliser certains caractères génériques (par exemple, **`prod-*`**) pour faire correspondre plusieurs espaces de noms (par exemple, `prod-deployment` et `prod-test`). Pour plus d'informations, consultez Caractères génériques de profils Fargate.

   b. (Facultatif) Ajoutez des étiquettes Kubernetes au sélecteur. Ajoutez-les spécifiquement à celui auquel les Pods de l'espace de noms spécifié doivent correspondre.

      - Vous pouvez ajouter l'étiquette **`infrastructure: fargate`** au sélecteur, afin que seuls les Pods de l'espace de noms spécifié qui présentent également l'étiquette `infrastructure: fargate` Kubernetes correspondent au sélecteur.

- Vous pouvez utiliser certains caractères génériques (par exemple, **`key?: value?`**) pour faire correspondre plusieurs espaces de noms (par exemple, `keya: valuea` et `keyb: valueb`). Pour plus d'informations, consultez Caractères génériques de profils Fargate.

c. Choisissez Suivant.

7. Sur la page Vérifier et créer, vérifiez les informations de votre profil Fargate et choisissez Créer.

Mettre à jour CoreDNS

Par défaut, CoreDNS est configuré pour s'exécuter sur l'infrastructure Amazon EC2 sur les clusters Amazon EKS. Si vous souhaitez uniquement exécuter vos Pods sur Fargate dans votre cluster, effectuez les étapes suivantes.

> **Note**
>
> Si vous avez créé votre cluster à l'aide de `eksctl` en utilisant l'option `--fargate`, vous pouvez passer directement à Étapes suivantes.

1. Créez un profil Fargate pour CoreDNS avec la commande suivante. Remplacez *`my-cluster`* par votre nom de cluster, *`111122223333`* par votre ID de compte, *`AmazonEKSFargatePodExecutionRole`* par le nom de votre rôle d'exécution de Pod, et *`0000000000000001`*, *`0000000000000002`*, et *`0000000000000003`* par les ID de vos sous-réseaux privés. Si vous n'avez pas de rôle d'exécution de Pod existant, vous devez d'abord en créer un.

   > **Important**
   >
   > L'ARN de rôle ne peut pas inclure un chemin d'accès autre que `/`. Par exemple, si le nom de votre rôle est `development/apps/my-role`, vous devez le remplacer par `my-role` lorsque vous spécifiez l'ARN du rôle. Le format de l'ARN de rôle doit être `arn:aws:iam::111122223333:role/role-name`.

```
aws eks create-fargate-profile \
```

```
    --fargate-profile-name coredns \
    --cluster-name my-cluster \
    --pod-execution-role-arn
 arn:aws:iam::111122223333:role/AmazonEKSFargatePodExecutionRole \
    --selectors namespace=kube-system,labels={k8s-app=kube-dns} \
    --subnets subnet-0000000000000001 subnet-0000000000000002
 subnet-0000000000000003
```

2. Exécutez la commande suivante pour supprimer l'annotation `eks.amazonaws.com/compute-type : ec2` des Pods CoreDNS.

```
kubectl patch deployment coredns \
    -n kube-system \
    --type json \
    -p='[{"op": "remove", "path": "/spec/template/metadata/annotations/
eks.amazonaws.com~1compute-type"}]'
```

Étapes suivantes

- Vous pouvez commencer à migrer vos applications existantes pour les exécuter sur Fargate avec le flux suivant.
  1. Créez un profil Fargate correspondant à l'espace de noms Kubernetes et aux étiquettes Kubernetes de votre application.
  2. Supprimez et recréez tous les Pods existants, afin qu'ils soient programmés sur Fargate. Par exemple, la commande suivante déclenche le déploiement de `coredns`. Vous pouvez modifier l'espace de noms et le type de déploiement pour mettre à jour vos Pods spécifiques.

```
kubectl rollout restart -n kube-system deployment coredns
```

- Déployez Répartition de la charge des applications sur Amazon EKS pour autoriser les objets d'entrée pour vos Pods s'exécutant sur Fargate.
- Vous pouvez utiliser l'Vertical Pod Autoscaler (VPA) pour paramétrer la bonne taille du processeur et de la mémoire de vos Pods Fargate, puis utiliser l'Horizontal Pod Autoscaler pour mettre à l'échelle ces Pods. Si vous souhaitez que le Vertical Pod Autoscaler redéploie automatiquement des Pods vers Fargate avec des combinaisons de CPU et de mémoire plus conséquentes, définissez le mode du Vertical Pod Autoscaler sur `Auto` ou `Recreate`. Ceci permet de garantir une fonctionnalité correcte. Pour plus d'informations, consultez la documentation Vertical Pod Autoscaler sur GitHub.

- Vous pouvez configurer le collecteur AWS Distro pour OpenTelemetry (ADOT) pour la surveillance des applications en suivant ces instructions.

AWS Fargate profil

> **Important**
>
> AWS Fargate avec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Avant de programmer des Pods sur Fargate dans votre cluster, vous devez définir au moins un profil Fargate qui spécifie quels Pods utilisent Fargate lorsqu'ils sont lancés.

En tant qu'administrateur, vous pouvez utiliser un profil Fargate pour déclarer les Pods à exécuter sur Fargate. Pour ce faire, vous pouvez utiliser les sélecteurs du profil. Vous pouvez ajouter jusqu'à cinq sélecteurs à chaque profil. Chaque sélecteur doit contenir un espace de noms. Le sélecteur peut également inclure des étiquettes. Le champ de label se compose de plusieurs paires clé-valeur facultatives. Les pods qui correspondent à un sélecteur sont programmés sur Fargate. Les pods sont associés à l'aide d'un espace de noms et des étiquettes spécifiées dans le sélecteur. Si un sélecteur d'espace de noms est défini sans étiquettes, Amazon EKS tente de programmer tous les Pods qui s'exécutent dans cet espace de noms sur Fargate en utilisant le profil. Si a to-be-scheduled Pod correspond à l'un des sélecteurs du profil Fargate, cela Pod est planifié sur Fargate.

Si un Pod correspond à plusieurs profils Fargate, vous pouvez spécifier quel profil un Pod utilise en ajoutant l'étiquette Kubernetes à la spécification du Pod : `eks.amazonaws.com/fargate-profile: my-fargate-profile`. Le Pod doit correspondre à un sélecteur dans ce profil pour être programmé sur Fargate. Les règles d'affinité/anti-affinité Kubernetes ne s'appliquent pas et ne sont pas nécessaires avec les Pods Amazon EKS Fargate.

Lorsque vous créez un profil Fargate, vous devez spécifier un rôle d'exécution de Pod. Ce rôle d'exécution concerne les composants Amazon EKS qui s'exécutent sur l'infrastructure Fargate utilisant le profil. Il est ajouté au Role Based Access Control (RBAC, contrôle d'accès basé sur les rôles) du cluster Kubernetes à des fins d'autorisation. Ainsi, le `kubelet` qui est exécuté sur l'infrastructure Fargate peut s'enregistrer dans votre cluster Amazon EKS et apparaître dans votre cluster comme un nœud. Le rôle d'exécution du Pod fournit également des autorisations IAM à l'infrastructure Fargate pour permettre un accès en lecture aux référentiels d'images Amazon ECR. Pour plus d'informations, consultez Rôle IAM d'exécution de Pod Amazon EKS.

Les profils Fargate ne peuvent pas être modifiés. Toutefois, vous pouvez créer un nouveau profil mis à jour pour remplacer un profil existant, puis supprimer l'original.

> **Note**
>
> Tous les Pods en cours d'exécution utilisant un profil Fargate seront arrêtés et mis en attente lorsque le profil sera supprimé.

Si tous les profils Fargate d'un cluster ont l'état `DELETING`, vous devez attendre que ce profil Fargate soit définitivement supprimé avant de pouvoir créer d'autres profils dans ce cluster.

Amazon EKS et Fargate répartissent les Pods sur chacun des sous-réseaux qui sont définis dans le profil Fargate. Cependant, vous risquez de vous retrouver avec une répartition inégale. Si vous avez besoin d'une répartition uniforme, utilisez deux profils Fargate. Une répartition uniforme est importante dans les scénarios où vous souhaitez déployer deux répliques sans aucune interruption de service. Nous vous recommandons de n'avoir qu'un seul sous-réseau pour chaque profil.

Composants de profil Fargate

Les composants suivants sont contenus dans un profil Fargate.

Rôle d'exécution du pod

Lorsque votre cluster est créé Pods AWS Fargate, celui `kubelet` qui s'exécute sur l'infrastructure Fargate doit appeler les API en votre AWS nom. Par exemple, il doit effectuer des appels pour extraire des images de conteneur à partir d'Amazon ECR. Pour ce faire, le rôle d'exécution de Pod Amazon EKS fournit les autorisations IAM.

Lorsque vous créez un profil Fargate, vous devez spécifier un rôle d'exécution de Pod à utiliser avec vos Pods. Ce rôle est ajouté au contrôle d'accès basé sur les rôles (RBAC) Kubernetes du cluster à des fins d'autorisation. Ainsi, le `kubelet` exécuté sur l'infrastructure Fargate peut s'enregistrer dans votre cluster Amazon EKS et apparaître dans votre cluster comme un nœud. Pour plus d'informations, consultez Rôle IAM d'exécution de Pod Amazon EKS.

Sous-réseaux

Les ID des sous-réseaux pour y lancer des Pods utilisent ce profil. Pour l'instant, les Pods fonctionnant sur Fargate n'ont pas d'adresse IP publique. Par conséquent, seuls les sous-réseaux privés sans route directe vers une passerelle Internet sont acceptés pour ce paramètre.

Sélecteurs

Les sélecteurs à faire correspondre pour que les Pods utilisent ce profil Fargate. Vous pouvez spécifier jusqu'à cinq sélecteurs dans un profil Fargate. Les sélecteurs intègrent les composants suivants :

- Espace de noms : vous devez spécifier un espace de noms pour un sélecteur. Le sélecteur ne correspond qu'aux Pods qui sont créés dans cet espace de noms. Vous pouvez toutefois créer plusieurs sélecteurs pour cibler plusieurs espaces de noms.
- Étiquettes –V ous pouvez éventuellement spécifier des étiquettes Kubernetes à faire correspondre pour le sélecteur. Le sélecteur ne correspond qu'aux Pods qui possèdent toutes les étiquettes spécifiées dans le sélecteur.

Caractères génériques de profils Fargate

En plus des caractères autorisés par Kubernetes, vous êtes autorisé à utiliser ***** et **?** dans les critères de sélection pour les espaces de noms, les clés d'étiquette et les valeurs d'étiquette :

- ***** représente aucun, un ou plusieurs caractères. Par exemple, **`prod*`** peut représenter `prod` et `prod-metrics`.
- **?** représente un caractère unique (par exemple, **`value?`** peut représenter `valuea`). Cependant, il ne peut pas représenter `value` et `value-a`, parce que ? ne peut représenter qu'un seul et unique caractère.

Ces caractères génériques peuvent être utilisés dans n'importe quelle position et en combinaison (par exemple, **`prod*`**, **`*dev`** et **`frontend*?`**). Les autres caractères génériques et formes de correspondance de modèles, tels que les expressions régulières, ne sont pas pris en charge.

S'il existe plusieurs profils correspondants pour l'espace de noms et les étiquettes dans la spécification du Pod, Fargate sélectionne le profil avec un tri alphanumérique par nom de profil. Par exemple, si le profil A (avec le nom `beta-workload`) et le profil B (avec le nom `prod-workload`) ont des sélecteurs correspondants pour les Pods à lancer, Fargate choisit le profil A (`beta-workload`) pour les Pods. Les Pods ont des étiquettes avec le profil A sur les Pods (par exemple, `eks.amazonaws.com/fargate-profile=beta-workload`).

Si vous souhaitez migrer des Pods Fargate existants vers de nouveaux profils qui utilisent des caractères génériques, vous pouvez procéder de deux manières :

- Créez un nouveau profil avec les sélecteurs correspondants, puis supprimez les anciens profils. Les pods étiquetés avec d'anciens profils sont reprogrammés vers de nouveaux profils correspondants.
- Si vous souhaitez migrer des charges de travail mais que vous ne savez pas quelles étiquettes Fargate figurent sur chaque Pod Fargate, vous pouvez utiliser la méthode suivante. Créez un nouveau profil avec un nom qui trie d'abord par ordre alphanumérique parmi les profils du même cluster. Recyclez ensuite les Pods Fargate qui doivent être migrés vers de nouveaux profils.

Création d'un profil Fargate

Cette rubrique décrit comment créer un profil Fargate. Vous devez également avoir créé un rôle d'exécution de Pod à utiliser pour votre profil Fargate. Pour plus d'informations, consultezRôle IAM d'exécution de Pod Amazon EKS. Podsqui s'exécutent sur Fargate ne sont pris en charge que sur les sous-réseaux privés dotés d'un accès Services AWS par passerelle NAT, mais ne constituent pas une route directe vers une passerelle Internet. Le VPC de votre cluster doit donc disposer de sous-réseaux privés. Vous pouvez créer un profil avec `eksctl` ou la AWS Management Console.

Cette procédure nécessite `eksctl` version `0.183.0` ou ultérieure. Vous pouvez vérifier votre version avec la commande suivante :

```
eksctl version
```

Pour les instructions d'installation ou de mise à niveau de `eksctl`, consultez la rubrique Installation dans la documentation `eksctl`.

eksctl

Pour créer un profil Fargate avec **`eksctl`**

Créez votre profil Fargate avec la commande `eksctl` suivante, en remplaçant chaque *`example value`* par vos propres valeurs. Vous devez spécifier un espace de noms. Cependant, l'option `--labels` n'est pas obligatoire.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

Vous pouvez utiliser certains caractères génériques pour les étiquettes *my-kubernetes-namespace* et *key=value*. Pour plus d'informations, consultez Caractères génériques de profils Fargate.

AWS Management Console

Pour créer un profil Fargate pour un cluster avec AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le cluster pour lequel vous voulez créer un profil Fargate.
3. Choisissez l'onglet Calcul.
4. Sous Fargate profiles (Profils Fargate), choisissez Add Fargate profile (Ajouter un profil Fargate).
5. Sur la page Configure Fargate profile (Configurer le profil Fargate), procédez comme suit :
   a. Pour Nom, saisissez un nom unique pour votre profil Fargate, tel que ***my-profile***.
   b. Pour Rôle d'exécution du pod, choisissez le rôle d'exécution du Pod à utiliser avec votre profil Fargate. Seuls les rôles IAM avec le principal de service `eks-fargate-pods.amazonaws.com` sont affichés. Si vous ne voyez aucun rôle répertorié ici, vous devez en créer un. Pour plus d'informations, consultez Rôle IAM d'exécution de Pod Amazon EKS.
   c. Modifiez les sous-réseaux sélectionnés selon vos besoins.

      > **Note**
      >
      > Seuls les sous-réseaux privés sont pris en charge pour les Pods qui s'exécutent sur Fargate.

   d. Dans Identifications, vous pouvez éventuellement étiqueter votre profil Fargate. Ces balises ne se propagent pas aux autres ressources qui sont associées au profil, comme les Pods.
   e. Choisissez Suivant.
6. Sur la page Configurer la sélection de Pod, procédez comme suit :
   a. Pour Espace de noms, saisissez un espace de noms correspondant aux Pods.

- Vous pouvez utiliser des espaces de noms spécifiques pour les faire correspondre, tels que **`kube-system`** ou **`default`**.
- Vous pouvez utiliser certains caractères génériques (par exemple, **`prod-*`**) pour faire correspondre plusieurs espaces de noms (par exemple, `prod-deployment` et `prod-test`). Pour plus d'informations, consultez Caractères génériques de profils Fargate.

b. (Facultatif) Ajoutez des étiquettes Kubernetes au sélecteur. Ajoutez-les spécifiquement à celui auquel les Pods de l'espace de noms spécifié doivent correspondre.

- Vous pouvez ajouter l'étiquette **`infrastructure: fargate`** au sélecteur, afin que seuls les Pods de l'espace de noms spécifié qui présentent également l'étiquette `infrastructure: fargate` Kubernetes correspondent au sélecteur.
- Vous pouvez utiliser certains caractères génériques (par exemple, **`key?: value?`**) pour faire correspondre plusieurs espaces de noms (par exemple, `keya: valuea` et `keyb: valueb`). Pour plus d'informations, consultez Caractères génériques de profils Fargate.

c. Choisissez Suivant.

7. Sur la page Vérifier et créer, vérifiez les informations de votre profil Fargate et choisissez Créer.

Suppression d'un profil Fargate

Cette rubrique décrit comment supprimer un profil Fargate.

Lorsque vous supprimez un profil Fargate, tous les Pods qui étaient programmés sur Fargate avec ce profil sont supprimés. Si ces Pods correspondent à un autre profil Fargate, ils sont programmés sur Fargate avec ce profil. S'ils ne correspondent plus à aucun profil Fargate, ils ne sont pas planifiés sur Fargate et peuvent rester en attente.

Un seul profil Fargate d'un cluster peut avoir le statut `DELETING` à la fois. Vous devez attendre que la suppression d'un profil Fargate soit terminée pour pouvoir supprimer un autre profil de ce cluster.

Vous pouvez supprimer un profil avec`eksctl`, le AWS Management Console, ou le AWS CLI. Sélectionnez l'onglet portant le nom de l'outil que vous souhaitez utiliser pour supprimer votre profil.

eksctl

Pour supprimer un profil Fargate avec **`eksctl`**

Utilisez la commande suivante pour supprimer un profil d'un cluster. Remplacez chaque *example value* par vos propres valeurs.

```
eksctl delete fargateprofile  --name my-profile --cluster my-cluster
```

AWS Management Console

Pour supprimer un profil Fargate d'un cluster avec AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters. Dans la liste des clusters, sélectionnez celui dont vous souhaitez supprimer le profil Fargate.
3. Choisissez l'onglet Calcul.
4. Sélectionnez le profil Fargate à supprimer, puis l'option Delete (Supprimer).
5. Sur la page Delete Fargate profile (Supprimer le profil Fargate), saisissez le nom du profil, puis sélectionnez Delete (Supprimer).

AWS CLI

Pour supprimer un profil Fargate avec AWS CLI

Utilisez la commande suivante pour supprimer un profil d'un cluster. Remplacez chaque *example value* par vos propres valeurs.

```
aws eks delete-fargate-profile --fargate-profile-name my-profile --cluster-name my-cluster
```

Configuration de Pod Fargate

> **Important**
>
> AWS Fargateavec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Cette section décrit certains des détails de configuration de Pod unique pour l'exécution de Pods Kubernetes sur AWS Fargate.

CPU et mémoire Pod

Grâce à Kubernetes, vous avez la possibilité de définir des demandes, une quantité minimale de vCPU et des ressources de mémoire allouées à chaque conteneur dans un Pod. Les Pods sont programmés par Kubernetes pour s'assurer qu'au moins les ressources demandées pour chaque Pod soient disponibles sur la ressource de calcul. Pour plus d'informations, consultez Gestion des ressources de calcul des conteneurs dans la documentation Kubernetes.

> **Note**
>
> Étant donné qu'Amazon EKS Fargate n'exécute qu'un seul Pod par nœud, le scénario d'expulsion des Pods en cas de diminution des ressources ne se produit pas. Tous les Pods Amazon EKS Fargate s'exécutent avec une priorité garantie, de sorte que le processeur et la mémoire demandés soient égaux à la limite pour tous les conteneurs. Pour plus d'informations, voir Configuration de la qualité de service des Pods dans la documentation Kubernetes.

Lorsque les Pods sont programmés sur Fargate, les réservations de vCPU et de mémoire dans la spécification du Pod déterminent la quantité de processeur et de mémoire à allouer au Pod.

- La demande maximale de tous les conteneurs Init est utilisée pour déterminer les besoins en vCPU et en mémoire de la demande Init.
- Les demandes de tous les conteneurs à longue durée d'exécution sont additionnées pour déterminer les besoins en vCPU et en mémoire de la demande à longue durée d'exécution.
- La plus grande des deux valeurs ci-dessus est choisie pour la demande de vCPU et la demande de mémoire à utiliser pour votre Pod.
- Fargate ajoute 256 Mo à la réservation de mémoire de chaque Pod pour les composants Kubernetes requis (`kubelet`, `kube-proxy` et `containerd`).

Fargate arrondit la configuration de calcul indiquée ci-dessous qui correspond le mieux à la somme des demandes de vCPU et les demandes de mémoire, afin de garantir que les Pods disposent toujours des ressources dont elles ont besoin pour s'exécuter.

Si vous ne spécifiez pas de combinaison vCPU et mémoire, la plus petite combinaison disponible est utilisée (0,25 vCPU et 0,5 Go de mémoire).

Le tableau suivant répertorie les combinaisons de vCPU et de mémoire qui sont disponibles pour les Pods qui s'exécutent sur Fargate.

Valeur vCPU	Valeur de mémoire
0,25 vCPU	0,5 Go, 1 Go, 2 Go
0,5 vCPU	1 Go, 2 Go, 3 Go, 4 Go
1 vCPU	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go
2 vCPU	Entre 4 Go et 16 Go par incrément de 1 Go
4 vCPU	Entre 8 Go et 30 Go par incrément de 1 Go
8 vCPU	Entre 16 Go et 60 Go par incréments de 4 Go
16 vCPU	Entre 32 Go et 120 Go par incréments de 8 Go

La mémoire supplémentaire réservée aux composants Kubernetes peut entraîner le provisionnement d'une tâche Fargate avec plus de vCPU que le nombre demandé. Par exemple, une demande pour 1 vCPU et 8 Go de mémoire verra 256 Mo ajoutés à sa demande de mémoire, et approvisionnera une tâche Fargate avec 2 vCPU et 9 Go de mémoire, puisqu'aucune tâche avec 1 vCPU et 9 Go de mémoire n'est disponible.

Il n'existe pas de corrélation entre la taille du Pod s'exécutant sur Fargate et la taille du nœud signalée par Kubernetes avec `kubectl get nodes`. La taille du nœud signalée est souvent supérieure à la capacité du Pod. Vous pouvez vérifier la capacité du Pod avec la commande suivante. Remplacez *`default`* par l'espace de noms de votre Pod et *`pod-name`* par le nom de votre Pod.

```
kubectl describe pod --namespace default pod-name
```

L'exemple qui suit illustre un résultat.

```
[...]
```

```
annotations:
    CapacityProvisioned: 0.25vCPU 0.5GB
[...]
```

L'annotation `CapacityProvisioned` représente la capacité du Pod appliquée et détermine le coût de votre Pod qui s'exécute sur Fargate. Pour plus d'informations sur la tarification des configurations de calcul, reportez-vous à Tarification AWS Fargate.

Stockage Fargate

Un Pod exécuté sur Fargate monte automatiquement un système de fichiers Amazon EFS. Vous ne pouvez pas utiliser l'approvisionnement dynamique des volumes persistants avec les nœuds Fargate, mais vous pouvez utiliser l'approvisionnement statique. Pour plus d'informations, consultez Amazon EFS CSI Driver activé GitHub.

Une fois provisionné, chaque Pod exécuté sur Fargate reçoit par défaut un espace de stockage éphémère de 20 GiB. Ce type de stockage est supprimé après l'arrêt du Pod. Lorsque de nouveaux Pods sont lancés sur Fargate, le chiffrement du volume de stockage éphémère est activé par défaut. Le stockage éphémère du Pod est chiffré avec un algorithme de chiffrement AES-256 utilisant des clés gérées par AWS Fargate.

> **Note**
>
> La capacité de stockage par défaut pour les Pods Amazon EKS exécutés sur Fargate est inférieure à 20 GiB. C'est parce qu'une partie de l'espace est utilisée par le `kubelet` et d'autres modules Kubernetes chargés dans le Pod.

Vous pouvez augmenter la capacité totale de stockage éphémère jusqu'à un maximum de 175 GiB. Pour configurer la taille avec Kubernetes, spécifiez les demandes de ressource `ephemeral-storage` pour chaque conteneur d'un Pod. Lorsque Kubernetes programme les Pods, le système s'assure que la somme des demandes de ressources pour chaque Pod est inférieure à la capacité de la tâche Fargate. Pour plus d'informations, voir Gestion des ressources des Pods et des conteneurs dans la documentation Kubernetes.

Amazon EKS Fargate alloue une capacité de stockage éphémère supérieure à celle demandée aux fins de l'utilisation du système. Par exemple, une demande de 100 GiB allouera à une tâche Fargate une capacité de stockage éphémère de 115 GiB.

Application de correctifs au système d'exploitation Fargate

> **Important**
>
> AWS Fargateavec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Amazon EKS doit régulièrement appliquer des correctifs au système d'exploitation pour les nœuds AWS Fargate pour assurer leur sécurité. Dans le cadre du processus d'application des correctifs, nous recyclons les nœuds pour installer les correctifs du système d'exploitation. Les tentatives de mises à jour sont réalisées de manière à avoir des répercussions minimales sur vos services. Toutefois, si les Pods ne sont pas expulsés avec succès, ils doivent parfois être supprimés. Voici les actions que vous pouvez effectuer pour minimiser les perturbations potentielles :

- Définissez des budgets d'interruption de Pod (PDB) appropriés pour contrôler le nombre de Pods simultanément hors service.
- Créez des EventBridge règles Amazon pour gérer les expulsions échouées avant qu'elles ne Pods soient supprimées.
- Créez une configuration de notification dans Notifications d'utilisateur AWS.

Amazon EKS travaille en étroite collaboration avec la communauté Kubernetes pour corriger les bugs et appliquer les correctifs de sécurité le plus rapidement possible. Tous les Pods Fargate démarrent la version de correctif Kubernetes la plus récente, disponible auprès d'Amazon EKS pour la version Kubernetes de votre cluster. Si un Pod présente une ancienne version de correctif, Amazon EKS peut le recycler pour le mettre à jour à la dernière version. Cela garantit que vos Pods incluent les dernières mises à jour de sécurité. De cette façon, en cas de problème critique inhérent aux vulnérabilités et expositions courantes (CVE), vous êtes tenu au courant afin de réduire les risques de sécurité.

Pour limiter le nombre de Pods hors service simultanément lors de l'application du recyclage des Pods, vous pouvez définir des budgets d'interruption de Pod (PDB). Vous pouvez utiliser les PDB pour définir une disponibilité minimale en fonction des exigences de chacune de vos applications, tout en autorisant les mises à jour. Pour plus d'informations, veuillez consulter Spécification d'un budget d'interruption pour votre application dans la documentation Kubernetes.

Amazon EKS utilise l'API d'expulsion pour drainer le Pod en toute sécurité tout en respectant les PDB que vous avez définis pour l'application. Les pods sont expulsés par zone de disponibilité pour minimiser les répercussions. Si l'expulsion réussit, le nouveau Pod reçoit le dernier correctif et aucune autre action n'est requise.

Lorsque l'expulsion d'un Pod échoue, Amazon EKS envoie un événement à votre compte avec des informations sur les Pods dont l'expulsion a échoué. Vous pouvez agir en fonction du message avant l'heure de fin prévue. Le temps spécifique varie en fonction de l'urgence du correctif. Le moment venu, Amazon EKS tente à nouveau d'expulser les Pods. Toutefois, si l'expulsion échoue lors de cette tentative, aucun nouvel événement n'est envoyé. Si l'expulsion échoue à nouveau, vos Pods existants sont supprimés périodiquement afin que les nouveaux Pods puissent bénéficier du dernier correctif.

Vous trouverez ci-dessous un exemple d'événement reçu lorsque l'expulsion d'un Pod échoue. Il contient des informations sur le cluster, le nom du Pod, l'espace de noms du Pod, le profil Fargate et l'heure de fin programmée.

```
{
    "version": "0",
    "id": "12345678-90ab-cdef-0123-4567890abcde",
    "detail-type": "EKS Fargate Pod Scheduled Termination",
    "source": "aws.eks",
    "account": "111122223333",
    "time": "2021-06-27T12:52:44Z",
    "region": "region-code",
    "resources": [
        "default/my-database-deployment"
    ],
    "detail": {
        "clusterName": "my-cluster",
        "fargateProfileName": "my-fargate-profile",
        "podName": "my-pod-name",
        "podNamespace": "default",
        "evictErrorMessage": "Cannot evict pod as it would violate the pod's disruption
 budget",
        "scheduledTerminationTime": "2021-06-30T12:52:44.832Z[UTC]"
    }
}
```

En outre, si plusieurs PDB sont associés à un Pod, un événement d'échec d'expulsion peut survenir. Cet événement renvoie le message d'erreur suivant.

```
"evictErrorMessage": "This pod has multiple PodDisruptionBudget, which the eviction
 subresource does not support",
```

Vous pouvez créer une action souhaitée en fonction de cet événement. Par exemple, vous pouvez ajuster votre budget d'interruption de Pod (PDB) pour contrôler l'expulsion des Pods. Plus précisément, imaginons que vous utilisez un PDB qui spécifie le pourcentage cible de Pods disponibles. Avant que la résiliation de vos Pods soit forcée lors d'une mise à niveau, vous pouvez ajuster le PDB à un pourcentage différent de Pods. Pour recevoir cet événement, vous devez créer une EventBridge règle Amazon dans le Compte AWS et Région AWS auquel appartient le cluster. La règle doit utiliser le modèle personnalisé suivant. Pour plus d'informations, consultez la section Création de EventBridge règles Amazon qui réagissent aux événements dans le guide de EventBridge l'utilisateur Amazon.

```
{
  "source": ["aws.eks"],
  "detail-type": ["EKS Fargate Pod Scheduled Termination"]
}
```

Une cible appropriée peut être définie pour que l'événement le capture. Pour obtenir la liste complète des cibles disponibles, consultez les EventBridge cibles Amazon dans le guide de EventBridge l'utilisateur Amazon. Vous pouvez également créer une configuration de notification dans Notifications d'utilisateurs AWS. Lorsque vous utilisez le AWS Management Console pour créer la notification, sous Règles d'événement, choisissez Elastic Kubernetes Service (EKS) pour Service AWS le nom et EKS Fargate Pod Scheduled Termination pour le type d'événement. Pour plus d'informations, consultez la section Prise en main des notifications d'utilisateur AWS dans le Guide de l'utilisateur des notifications d'utilisateur AWS.

Métriques Fargate

> **Important**
>
> AWS Fargate avec Amazon EKS n'est pas disponible dans AWS GovCloud (USA Est) et AWS GovCloud (USA-Ouest).

Vous pouvez collecter des métriques système et des mesures d'utilisation CloudWatch pour AWS Fargate.

Métriques d'application

Pour les applications exécutées sur Amazon EKS et AWS Fargate, vous pouvez utiliser AWS Distro pour OpenTelemetry (ADOT). ADOT vous permet de collecter des métriques système et de les envoyer aux tableaux de bord CloudWatch Container Insights. Pour démarrer avec ADOT pour les applications exécutées sur Fargate, consultez Utilisation de CloudWatch Container Insights avec AWS Distro pour OpenTelemetry dans la documentation ADOT.

Métriques d'utilisation

Vous pouvez utiliser les métriques d'utilisation CloudWatch pour fournir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces métriques pour visualiser l'utilisation actuelle de vos services sur les graphiques et tableaux de bord CloudWatch.

Les métriques d'utilisation AWS Fargate correspondent aux Service Quotas AWS. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche un quota de service. Pour de plus amples informations sur les Service Quotas pour Fargate, consultez Service quotas Amazon EKS.

AWS Fargate publie les métriques suivantes dans l'espace de noms `AWS/Usage`.

Métrique	Description
`ResourceCount`	Nombre total des ressources spécifiées exécutées sur votre compte. La ressource est définie par les dimensions associées à la métrique.

Les dimensions suivantes permettent d'affiner les métriques d'utilisation publiées par AWS Fargate.

Dimension	Description
`Service`	Nom du service AWS contenant la ressource. Pour les métriques d'utilisation d'AWS Fargate, la valeur de cette dimension est `Fargate`.
`Type`	Type d'entité faisant l'objet d'un rapport. Actuellement, la seule valeur valide pour les métriques d'utilisation d'AWS Fargate est `Resource`.

Dimension	Description
`Resource`	Type de ressource en cours d'exécution. Actuellement, AWS Fargate renvoie des informations sur votre utilisation de Fargate On-Demand. La valeur de ressource pour l'utilisation de Fargate On-Demand est `OnDemand`. **Note** L'utilisation de Fargate à la demande combine les Pods Amazon EKS utilisant Fargate, les tâches Amazon ECS utilisant le type de lancement Fargate et les tâches Amazon ECS utilisant le fournisseur de capacité `FARGATE`.
`Class`	Classe de ressource suivie. Actuellement, AWS Fargate n'utilise pas la dimension Class.

Création d'une alarme CloudWatch pour contrôler les métriques d'utilisation des ressources Fargate

AWS Fargate fournit des métriques d'utilisation CloudWatch qui correspondent aux quotas de service AWS pour l'utilisation des ressources Fargate On-Demand. Dans la console Service Quotas, vous pouvez visualiser votre utilisation sur un graphique. Vous pouvez également configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour plus d'informations, consultez Métriques Fargate.

Suivez les étapes ci-dessous pour créer une alarme CloudWatch basée sur l'une des métriques d'utilisation Fargate.

Pour créer une alarme basée sur vos quotas d'utilisation de Fargate (AWS Management Console)

1. Ouvrez la console Service Quotas à l'adresse https://console.aws.amazon.com/servicequotas/.
2. Dans le panneau de navigation de gauche, sélectionnez Services AWS.
3. Dans la liste des services AWS, recherchez et sélectionnez AWS Fargate.
4. Dans la liste Quotas de service, sélectionnez le quota d'utilisation Fargate pour lequel vous souhaitez créer une alarme.

5. Dans la section Amazon CloudWatch alarms (Alarmes Amazon CloudWatch), sélectionnez Create (Créer).
6. Pour Alarm threshold (Seuil d'alarme), choisissez le pourcentage de la valeur de quota appliquée que vous souhaitez définir comme valeur d'alarme.
7. Pour Nom de l'alarme, saisissez un nom pour l'alarme, puis choisissez Créer.

Journalisation Fargate

> **Important**
>
> AWS Fargate avec Amazon EKS n'est pas disponible en AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

Amazon EKS on Fargate propose un routeur de journal intégré basé sur Fluent Bit. Cela signifie que vous n'exécutez pas explicitement un conteneur Fluent Bit comme composant complémentaire, mais qu'Amazon l'exécute pour vous. Tout ce que vous avez à faire est de configurer le routeur de journaux. La configuration se fait par le biais d'un `ConfigMap` qui doit répondre aux critères suivants :

- Nommé `aws-logging`
- Créé dans un espace de noms dédié appelé `aws-observability`
- Ne doit pas dépasser 5 300 caractères.

Une fois que vous avez créé le `ConfigMap`, Amazon EKS on Fargate le détecte automatiquement et configure le routeur de journaux avec lui. Fargate utilise une version de Fluent Bit for, une distribution Fluent Bit conforme en amont AWS de managed by. AWS Pour plus d'informations, reportez-vous AWS à la section Fluent Bit sur GitHub.

Le routeur de journaux vous permet d'utiliser l'ensemble des services proposés AWS pour l'analyse et le stockage des journaux. Vous pouvez diffuser les journaux de Fargate directement vers Amazon CloudWatch, Amazon Service. OpenSearch Vous pouvez également diffuser des journaux vers des destinations telles qu'Amazon S3, Amazon Kinesis Data Streams et des outils partenaires via Amazon Data Firehose.

Prérequis

- Un profil Fargate existant qui spécifie un espace de noms Kubernetes existant dans lequel vous déployez des Pods Fargate. Pour de plus amples informations, veuillez consulter Création d'un profil Fargate pour votre cluster.
- Un rôle d'exécution de Pod Fargate existant. Pour de plus amples informations, veuillez consulter Création d'un rôle d'exécution de Pod Fargate.

Configuration du routeur de journaux

Pour configurer le routeur de journaux

Dans les étapes suivantes, remplacez chaque *example value* par vos propres valeurs.

1. Créez un espace de noms Kubernetes dédié nommé `aws-observability`.

   a. Enregistrez le contenu suivant dans un fichier nommé *aws-observability-namespace*`.yaml` sur votre ordinateur. La valeur pour `name` doit être `aws-observability` et le label `aws-observability: enabled` est obligatoire.

   ```
   kind: Namespace
   apiVersion: v1
   metadata:
     name: aws-observability
     labels:
       aws-observability: enabled
   ```

   b. Créez l'espace de noms.

   ```
   kubectl apply -f aws-observability-namespace.yaml
   ```

2. Créez un `ConfigMap` avec une valeur de données `Fluent Conf` pour envoyer les journaux des conteneurs vers une destination. Fluent Conf est Fluent Bit, qui est un langage de configuration de processeur de journaux rapide et léger, utilisé pour acheminer les journaux des conteneurs vers une destination de journalisation de votre choix. Pour plus d'informations, consultez Fichier de configuration dans la documentation Fluent Bit.

> **Important**
>
> Dans une `Fluent Conf` type, les principales sections incluses sont `Service`, `Input`, `Filter` et `Output`. Le routeur de journaux Fargate n'accepte cependant que :
>
> - Les sections `Filter` et `Output`.
> - Une section `Parser`.
>
> Si vous fournissez d'autres sections, elles seront rejetées.

Le routeur de journal Fargate gère les sections `Service` et `Input`. Il possède la section `Input` suivante, qui ne peut pas être modifiée et n'est pas nécessaire dans votre `ConfigMap`. Cependant, vous pouvez en tirer des informations, telles que la limite de la mémoire tampon et la balise appliquée pour les journaux.

```
[INPUT]
    Name tail
    Buffer_Max_Size 66KB
    DB /var/log/flb_kube.db
    Mem_Buf_Limit 45MB
    Path /var/log/containers/*.log
    Read_From_Head On
    Refresh_Interval 10
    Rotate_Wait 30
    Skip_Long_Lines On
    Tag kube.*
```

Lors de la création du `ConfigMap`, prenez en compte les règles suivantes que Fargate utilise pour valider les champs :

- `[FILTER]`, `[OUTPUT]` et `[PARSER]` sont censés être spécifiés sous chaque clé correspondante. Par exemple, `[FILTER]` doit être inférieur à `filters.conf`. Vous pouvez avoir un ou plusieurs `[FILTER]` sous `filters.conf`. `[OUTPUT]` et `[PARSER]` doivent également être sous leurs clés correspondantes. En spécifiant plusieurs sections `[OUTPUT]`, vous pouvez acheminer vos journaux vers différentes destinations en même temps.

- Fargate valide les clés requises de chaque section. `Name` et `match` sont nécessaires pour chaque `[FILTER]` et `[OUTPUT]`. `Name` et `format` sont nécessaires pour chaque `[PARSER]`. Ces noms sont sensibles à la casse.
- Les variables d'environnement telles que `${ENV_VAR}` ne sont pas autorisées dans le `ConfigMap`.
- L'indentation doit être la même pour la directive ou la paire clé-valeur dans chaque `filters.conf`, `output.conf` et `parsers.conf`. Les paires clé-valeur doivent être indentées plus que les directives.
- Fargate valide par rapport aux filtres pris en charge suivants : `grep`, `parser`, `record_modifier`, `rewrite_tag`, `throttle`, `nest`, `modify` et `kubernetes`.
- Fargate valide par rapport à la sortie prise en charge suivante : `es`, `firehose`, `kinesis_firehose`, `cloudwatch`, `cloudwatch_logs` et `kinesis`.
- Au moins un plugin `Output` doit être fourni dans le `ConfigMap` pour activer la journalisation. `Filter` et `Parser` ne sont pas nécessaires pour activer la journalisation.

Vous pouvez également exécuter Fluent Bit sur Amazon EC2 en utilisant la configuration souhaitée pour résoudre les problèmes qui surviennent lors de la validation. Créez votre `ConfigMap` en utilisant l'un des exemples suivants.

> **Important**
>
> La journalisation Amazon EKS Fargate ne prend pas en charge la configuration dynamique de `ConfigMaps`. Les modifications apportées à `ConfigMaps` sont appliquées uniquement aux nouveaux Pods. Les modifications ne sont pas appliquées aux Pods existants.

Créez un `ConfigMap` en utilisant l'exemple pour votre destination de journal désirée.

> **Note**
>
> Vous pouvez également utiliser Amazon Kinesis Data Streams comme destination du journal. Si vous utilisez Kinesis Data Streams, assurez-vous que l'autorisation `kinesis:PutRecords` a été accordée au rôle d'exécution du pod. Pour

plus d'informations, consultez Permissions d'Amazon Kinesis Data Streams dans le Manuel officiel Fluent Bit.

CloudWatch

Pour créer une **`ConfigMap`** pour CloudWatch

Deux options de sortie s'offrent à vous lorsque vous utilisez CloudWatch :

- Un plugin de sortie écrit en C
- Un plugin de sortie écrit en Golang

L'exemple suivant montre comment utiliser le `cloudwatch_logs` plugin pour envoyer des journaux à CloudWatch.

1. Enregistrez le contenu suivant dans un fichier nommé *`aws-logging-cloudwatch-configmap`*`.yaml`. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Les paramètres sous `[OUTPUT]` sont requises.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  flb_log_cw: "false"  # Set to true to ship Fluent Bit process logs to
 CloudWatch.
  filters.conf: |
    [FILTER]
        Name parser
        Match *
        Key_name log
        Parser crio
    [FILTER]
        Name kubernetes
        Match kube.*
        Merge_Log On
        Keep_Log Off
        Buffer_Size 0
        Kube_Meta_Cache_TTL 300s
```

```
  output.conf: |
    [OUTPUT]
        Name cloudwatch_logs
        Match   kube.*
        region region-code
        log_group_name my-logs
        log_stream_prefix from-fluent-bit-
        log_retention_days 60
        auto_create_group true
  parsers.conf: |
    [PARSER]
        Name crio
        Format Regex
        Regex ^(?<time>[^ ]+) (?<stream>stdout|stderr) (?<logtag>P|F) (?
<log>.*)$
        Time_Key    time
        Time_Format %Y-%m-%dT%H:%M:%S.%L%z
```

2. Appliquez le manifeste à votre cluster.

```
kubectl apply -f aws-logging-cloudwatch-configmap.yaml
```

3. Téléchargez la politique CloudWatch IAM sur votre ordinateur. Vous pouvez également consulter la politique sur GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/cloudwatchlogs/permissions.json
```

Amazon OpenSearch Service

Pour créer un OpenSearch service **ConfigMap** pour Amazon

Si vous souhaitez envoyer des journaux à Amazon OpenSearch Service, vous pouvez utiliser es output, qui est un plugin écrit enC. L'exemple suivant montre comment utiliser le plugin pour envoyer des journaux à OpenSearch.

1. Enregistrez le contenu suivant dans un fichier nommé *aws-logging-opensearch-configmap*.yaml. Remplacez chaque *example value* par vos propres valeurs.

```
kind: ConfigMap
apiVersion: v1
```

```
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
      Name  es
      Match *
      Host  search-example-gjxdcilagiprbglqn42jsty66y.region-
code.es.amazonaws.com
      Port  443
      Index example
      Type  example_type
      AWS_Auth On
      AWS_Region region-code
      tls   On
```

2. Appliquez le manifeste à votre cluster.

```
kubectl apply -f aws-logging-opensearch-configmap.yaml
```

3. Téléchargez la politique OpenSearch IAM sur votre ordinateur. Vous pouvez également consulter la politique sur GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-
fluent-logging-examples/mainline/examples/fargate/amazon-elasticsearch/
permissions.json
```

   Assurez-vous que le contrôle d'accès OpenSearch des tableaux de bord est correctement configuré. `all_access role`Dans les OpenSearch tableaux de bord, le rôle d'exécution Pod Fargate et le rôle IAM doivent être mappés. Le même mappage doit être fait pour le rôle `security_manager`. Vous pouvez ajouter les mappages précédents en sélectionnant `Menu`, `Security` et `Roles`, puis sélectionner les rôles correspondants. Pour plus d'informations, consultez Comment résoudre les problèmes liés aux CloudWatch journaux afin qu'ils soient diffusés sur mon domaine Amazon ES ? .

Firehose

Pour créer un **`ConfigMap`** pour Firehose

Deux options de sortie s'offrent à vous lorsque vous envoyez des logs à Firehose :

- `kinesis_firehose` : un plugin de sortie écrit en C.
- `firehose` : un plugin de sortie écrit en Golang.

L'exemple suivant vous montre comment utiliser le `kinesis_firehose` plugin pour envoyer des logs à Firehose.

1. Enregistrez le contenu suivant dans un fichier nommé *`aws-logging-firehose-configmap`*`.yaml`. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
     Name  kinesis_firehose
     Match *
     region region-code
     delivery_stream my-stream-firehose
```

2. Appliquez le manifeste à votre cluster.

```
kubectl apply -f aws-logging-firehose-configmap.yaml
```

3. Téléchargez la politique IAM de Firehose sur votre ordinateur. Vous pouvez également consulter la politique sur GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/kinesis-firehose/permissions.json
```

3. Créez une politique IAM depuis le fichier de politique que vous avez téléchargé à l'étape précédente.

```
aws iam create-policy --policy-name eks-fargate-logging-policy --policy-document
 file://permissions.json
```

4. Attachez la politique IAM au rôle d'exécution de pod spécifié pour votre profil Fargate avec la commande suivante. Remplacez *111122223333* par votre ID de compte. Remplacez *AmazonEKSFargatePodExecutionRole* par le rôle d'exécution de votre Pod (pour plus d'informations, veuillez consulter la rubrique Création d'un rôle d'exécution de Pod Fargate).

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/eks-fargate-logging-policy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Support de filtre Kubernetes

Cette fonction requiert la version Kubernetes minimale et le niveau de plateforme suivants, ou plus.

Version de Kubernetes	Niveau de la plateforme
1.23 et versions ultérieures	eks.1

Le filtre Kubernetes Fluent Bit vous permet d'ajouter des métadonnées Kubernetes à vos fichiers journaux. Pour plus d'informations sur le filtre, consultez Kubernetes dans la documentation Fluent Bit. Vous pouvez appliquer un filtre en utilisant le point de terminaison du serveur d'API.

```
filters.conf: |
    [FILTER]
        Name                kubernetes
        Match               kube.*
        Merge_Log           On
        Buffer_Size         0
        Kube_Meta_Cache_TTL 300s
```

Important

- `Kube_URL`, `Kube_CA_File`, `Kube_Token_Command` et `Kube_Token_File` sont des paramètres de configuration appartenant au service et ne doivent pas être spécifiés. Amazon EKS Fargate remplit ces valeurs.
- `Kube_Meta_Cache_TTL` est le temps pendant lequel Fluent Bit attend jusqu'à ce qu'il communique avec le serveur d'API pour obtenir les dernières métadonnées. Si la valeur

`Kube_Meta_Cache_TTL` n'est pas spécifiée, Amazon EKS Fargate ajoute une valeur par défaut de 30 minutes pour diminuer la charge sur le serveur d'API.

Pour envoyer les journaux de processus Fluent Bit à votre compte

Vous pouvez éventuellement envoyer des journaux de Fluent Bit processus à Amazon CloudWatch en utilisant les méthodes suivantes`ConfigMap`. L'expédition des journaux de traitement Fluent Bit CloudWatch nécessite des frais d'ingestion et de stockage supplémentaires. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
  labels:
data:
  # Configuration files: server, input, filters and output
  # ======================================================
  flb_log_cw: "true"  # Ships Fluent Bit process logs to CloudWatch.

  output.conf: |
    [OUTPUT]
        Name cloudwatch
        Match kube.*
        region region-code
        log_group_name fluent-bit-cloudwatch
        log_stream_prefix from-fluent-bit-
        auto_create_group true
```

Les journaux se trouvent dans Région AWS le répertoire sous lequel réside le cluster CloudWatch. Le nom du groupe de journaux est *`my-cluster`*`-fluent-bit-logs` et le nom du flux de journaux Fluent Bit est `fluent-bit-`*`podname`*`-`*`pod-namespace`*.

Note

- Les journaux de processus sont uniquement envoyés lorsque le processus Fluent Bit démarre avec succès. En cas d'échec lors du démarrage de Fluent Bit, les journaux

de processus sont manqués. Vous ne pouvez expédier les journaux de processus qu'à CloudWatch.

- Pour déboguer l'envoi des journaux de processus à votre compte, vous pouvez appliquer la `ConfigMap` précédente pour obtenir les journaux de processus. L'échec de démarrage de Fluent Bit est généralement dû au fait que votre `ConfigMap` n'est pas analysée ou acceptée par Fluent Bit lors du démarrage.

Pour arrêter l'envoi des journaux de processus Fluent Bit

Fluent BitLe processus d'expédition entraîne des CloudWatch coûts supplémentaires d'ingestion et de stockage des journaux. Pour exclure les journaux de processus d'une configuration de `ConfigMap` existante, procédez comme suit.

1. Localisez le groupe de CloudWatch journaux créé automatiquement pour les journaux de Fluent Bit processus de votre cluster Amazon EKS après avoir activé la journalisation Fargate. Il utilise le format `{cluster_name}-fluent-bit-logs`.
2. Supprimez les flux de CloudWatch journaux existants créés pour chaque CloudWatch journal de Pod's processus du groupe de journaux.
3. Modifiez la `ConfigMap` et définissez `flb_log_cw: "false"`.
4. Redémarrez tous les Pods existants du cluster.

Tester l'application

1. Déployez un exemple de Pod.
   a. Enregistrez le contenu suivant dans un fichier nommé *`sample-app`*`.yaml` sur votre ordinateur.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  namespace: same-namespace-as-your-fargate-profile
spec:
  replicas: 3
  selector:
    matchLabels:
```

```
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - name: http
              containerPort: 80
```

b. Appliquez le fichier manifeste à votre cluster.

```
kubectl apply -f sample-app.yaml
```

2. Affichez les journaux NGINX en utilisant les destinations que vous avez configurées dans le fichier `ConfigMap`.

Considérations sur les tailles

Nous vous suggérons de prévoir jusqu'à 50 Mo de mémoire pour le routeur de journaux. Si votre application doit générer des journaux à un débit très élevé, vous devez prévoir jusqu'à 100 Mo.

Résolution des problèmes

Pour confirmer que la fonctionnalité de journalisation est activée ou désactivée pour une quelconque raison, telle qu'un `ConfigMap` non valide, et déterminer la raison pour laquelle il n'est pas valide, vérifiez les événements de votre Pod avec **`kubectl describe pod`** ***`pod_name`***. La sortie peut inclure des événements de Pod qui précisent si la journalisation est activée ou non, comme l'exemple de sortie suivant.

```
[...]
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
                      kubernetes.io/psp: eks.privileged
[...]
Events:
  Type     Reason           Age        From
              Message
```

```
  ----     ------           ----       ----
                -------
  Warning  LoggingDisabled  <unknown>  fargate-scheduler
                Disabled logging because aws-logging configmap was not found. configmap
"aws-logging" not found
```

Les événements du Pod sont éphémères avec une période de temps dépendant des paramètres. Vous pouvez également afficher les annotations d'un Pod's à l'aide de **`kubectl describe pod pod-name`**. Dans l'annotation du Pod, il existe des informations sur l'activation ou la désactivation de la fonctionnalité de journalisation et la cause.

Choix d'un type d'instance Amazon EC2

Amazon EC2 fournit un large choix de types d'instance pour les composants master. Chaque type d'instance offre des capacités de calcul, de mémoire, de stockage et de réseau différentes. Chaque instance est également regroupée dans une famille d'instances en fonction de ces capacités. Pour obtenir une liste, consultez les types d'instances disponibles dans le guide de l'utilisateur Amazon EC2 et les types d'instances disponibles dans le guide de l'utilisateur Amazon EC2. Amazon EKS publie plusieurs variantes d'AMI Amazon EC2 pour permettre la prise en charge. Pour vous assurer que le type d'instance que vous sélectionnez est compatible avec Amazon EKS, tenez compte des critères suivants.

- Toutes les AMI Amazon EKS ne prennent actuellement pas en charge les familles `g5g` et `mac`.
- Les AMI Arm et Amazon EKS non accélérées ne prennent pas en charge les familles `g3`, `g4`, `inf` et `p`.
- Les AMI Amazon EKS accélérées ne prennent pas en charge les familles `a`, `c`, `hpc`, `m` et `t`.
- Pour les instances basées sur ARM, Amazon Linux 2023 (AL2023) ne prend en charge que les types d'instances qui utilisent des processeurs Graviton2 ou des processeurs ultérieurs. AL2023 ne prend pas en charge les `A1` instances.

Lorsque vous choisissez entre les types d'instances pris en charge par Amazon EKS, tenez compte des fonctionnalités suivantes de chaque type.

Nombre d'instances dans un groupe de nœuds

> En général, il est préférable d'utiliser un nombre restreint de grandes instances, surtout si vous disposez de beaucoup de Daemonsets. Chaque instance nécessitant des appels API vers le

serveur d'API, plus le nombre d'instances est élevé, plus la charge sur le serveur d'API est importante.

Système d'exploitation

Examinez les types d'instances pris en charge pour Linux, Windows et Bottlerocket. Avant de créer des instances Windows, consultez Activation de la prise en charge de Windows pour votre cluster Amazon EKS.

Architecture matérielle

Avez-vous besoin de x86 ou Arm ? Vous ne pouvez déployer que Linux sur Arm. Avant de déployer des instances Arm, consultez AMI Amazon Linux Arm optimisées pour Amazon EKS. Avez-vous besoin d'instances reposant sur le système Nitro System (Linux ou Windows) ou qui ont des fonctionnalités accélérées ? Si vous avez besoin de capacités accélérées, vous ne pouvez utiliser Linux qu'avec Amazon EKS.

Nombre maximal d'Pods

Étant donné que chaque Pod se voit attribuer sa propre adresse IP, le nombre d'adresses IP prises en charge par un type d'instance est un facteur qui détermine le nombre de Pods pouvant fonctionner sur l'instance. Pour déterminer manuellement le nombre de Pods pris en charge par un type d'instance, consultez Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2.

> **Note**
>
> Si vous utilisez une AMI Amazon Linux 2 optimisée pour Amazon EKS `v20220406` ou plus récent, vous pouvez utiliser un nouveau type d'instance sans mettre à niveau vers la dernière AMI. Pour ces AMI, l'AMI calcule automatiquement la valeur `max-pods` nécessaire si elle n'est pas répertoriée dans le fichier `eni-max-pods.txt`. Les types d'instance actuellement en prévisualisation peuvent ne pas être pris en charge par Amazon EKS par défaut. Les valeurs pour `max-pods` pour de tels types doivent encore être ajoutées à `eni-max-pods.txt` dans notre AMI.

AWS Les types d'instances Nitro System prennent éventuellement en charge un plus grand nombre d'adresses IP que les types d'instances autres que Nitro System. Cependant, toutes les adresses IP attribuées à une instance ne sont pas disponibles pour les Pods. Pour attribuer un nombre significativement plus élevé d'adresses IP à vos instances, la version `1.9.0` ou ultérieure du module complémentaire Amazon VPC CNI doit être installée dans votre cluster et configurée

de manière appropriée. Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2. Pour attribuer le plus grand nombre d'adresses IP à vos instances, vous devez avoir installé la version `1.10.1` ou une version ultérieure du module complémentaire Amazon VPC CNI dans votre cluster et déployer le cluster avec la famille `IPv6`.

Famille d'IP

Vous pouvez utiliser n'importe quel type d'instance pris en charge lorsque vous utilisez la famille `IPv4` pour un cluster, ce qui permet à votre cluster d'attribuer des adresses `IPv4` privées à vos Pods et services. Mais si vous souhaitez utiliser la famille `IPv6` pour votre cluster, alors vous devez utiliser les types d'instance AWS Nitro System ou les types d'instance matériel nu. Seul `IPv4` est pris en charge pour les instances Windows. Votre cluster doit utiliser la version `1.10.1` ou une version ultérieure du module complémentaire Amazon VPC CNI. Pour plus d'informations sur l'utilisation de `IPv6`, consultez IPv6adresses pour les clustersPods, et services.

Version du module complémentaire Amazon VPC CNI que vous exécutez

La dernière version du plug-in CNI Amazon VPC pour Kubernetes prend en charge ces types d'instance. Il peut être nécessaire de mettre à jour la version de votre module complémentaire CNI Amazon VPC pour bénéficier des derniers types d'instance pris en charge. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS. La dernière version prend en charge les dernières fonctions pour une utilisation avec Amazon EKS. Les versions antérieures ne prennent pas en charge toutes les fonctions. Vous pouvez afficher les fonctions prises en charge par les différentes versions dans Changelog sur GitHub.

Région AWS dans lequel vous créez vos nœuds

Tous les types d'instance ne sont pas toutes disponibles dans toutes les Régions AWS.

Si vous utilisez des groupes de sécurité pour les Pods

Si vous utilisez des groupes de sécurité pour les Pods, seuls des types d'instance spécifiques sont pris en charge. Pour plus d'informations, consultez Groupes de sécurité pour Pods.

Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2

Étant donné que chaque Pod se voit attribuer sa propre adresse IP, le nombre d'adresses IP prises en charge par un type d'instance est un facteur qui détermine le nombre de Pods pouvant

fonctionner sur l'instance. Amazon EKS fournit un script que vous pouvez télécharger et exécuter pour déterminer le nombre maximum de Pods recommandé par Amazon EKS à exécuter sur chaque type d'instance. Le script utilise les attributs matériels de chaque instance et les options de configuration pour déterminer le nombre maximum de Pods. Vous pouvez utiliser le nombre renvoyé dans ces étapes pour activer des capacités telles que l'attribution d'adresses IP aux Pods d'un sous-réseau différent de celui de l'instance et l'augmentation significative du nombre d'adresses IP pour votre instance. Si vous utilisez un groupe de nœuds géré composé de plusieurs types d'instances, utilisez une valeur qui fonctionne pour tous les types d'instances.

1. Téléchargez un script que vous pouvez utiliser pour calculer le nombre maximum de Pods pour chaque type d'instance.

```
curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/templates/al2/runtime/max-pods-calculator.sh
```

2. Marquez le script comme exécutable sur votre ordinateur.

```
chmod +x max-pods-calculator.sh
```

3. Exécutez le script en remplaçant *m5.large* par le type d'instance que vous prévoyez de déployer et *1.9.0-eksbuild.1* par votre version du module complémentaire CNI Amazon VPC. Pour déterminer la version de votre module complémentaire, consultez les procédures de mise à jour dans Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

```
./max-pods-calculator.sh --instance-type m5.large --cni-version 1.9.0-eksbuild.1
```

   L'exemple qui suit illustre un résultat.

```
29
```

   Vous pouvez ajouter les options suivantes au script pour connaître le nombre maximum de Pods pris en charge lors de l'utilisation des capacités facultatives.

   - `--cni-custom-networking-enabled` : utilisez cette option lorsque vous souhaitez attribuer des adresses IP provenant d'un sous-réseau différent de celui de votre instance. Pour plus d'informations, consultez Mise en réseau personnalisée pour les pods. L'ajout de cette option au script précédent avec les mêmes valeurs d'exemple donne `20`.
   - `--cni-prefix-delegation-enabled` : utilisez cette option lorsque vous souhaitez attribuer beaucoup plus d'adresses IP à chaque interface réseau Elastic. Cette fonctionnalité nécessite

une instance Amazon Linux qui fonctionne sur le système Nitro et la version `1.9.0` ou ultérieure du module complémentaire CNI Amazon VPC. Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2. L'ajout de cette option au script précédent avec les mêmes valeurs d'exemple donne `110`.

Vous pouvez également exécuter le script avec l'option `--help` permettant de voir toutes les options disponibles.

> **Note**
>
> Le script de calcul du nombre maximal de Pods limite la valeur renvoyée à `110` en fonction des seuils de capacité de mise à l'échelle de Kubernetes et des paramètres recommandés. Si votre type d'instance comporte plus de 30 vCPUs, cette limite passe à `250`, un chiffre basé sur des tests internes de l'équipe pour la capacité de mise à l'échelle d'Amazon EKS. Pour plus d'informations, consultez l'article de blog Plugin CNI Amazon VPC augmente les limites du nombre de pods par nœud (français non garanti).

AMI optimisées pour Amazon EKS

Vous pouvez déployer des nœuds avec des Amazon Machine Images (AMI) préconfigurées optimisées pour Amazon EKS, ou vos propres AMI personnalisées. Pour plus d'informations sur chaque type d'AMI optimisées pour Amazon EKS, veuillez consulter l'une des rubriques suivantes. Pour obtenir des instructions sur la façon de créer votre propre AMI personnalisée, veuillez consulter Script de création d'AMI Amazon Linux optimisées pour Amazon EKS.

Rubriques

- Amazon EKS a mis fin à la prise en charge de Dockershim
- AMI Amazon Linux optimisées pour Amazon EKS
- AMI Bottlerocket optimisées pour Amazon EKS
- AMI Ubuntu Linux optimisées pour Amazon EKS
- AMI Windows optimisées pour Amazon EKS

Amazon EKS a mis fin à la prise en charge de `Dockershim`

Kubernetes ne prend plus en charge `Dockershim`. L'équipe Kubernetes a supprimé l'environnement d'exécution dans Kubernetes version `1.24`. Pour de plus amples informations, veuillez consulter Kubernetes is Moving on From Dockershim: Commitments and Next Steps sur le Blog Kubernetes.

Amazon EKS a également mis fin à la prise en charge de `Dockershim` à partir de la version `1.24` de Kubernetes. Les AMI Amazon EKS officiellement publiées incluent `containerd` comme seul environnement d'exécution à partir de la version `1.24`. Cette rubrique couvre certains détails, mais de plus amples informations sont disponibles dans le guide Tout ce que vous devez savoir sur la migration vers containerd sur Amazon EKS.

Vous pouvez utiliser un plug-in `kubectl` pour identifier les charges de travail Kubernetes qui montent le volume du socket Docker. Pour de plus amples informations, veuillez consulter Détecteur pour Docker Socket (DDS) sur GitHub. Les AMI Amazon EKS qui exécutent des versions de Kubernetes antérieures à la version `1.24` utilisent Docker comme environnement d'exécution par défaut. Toutefois, ces AMI Amazon EKS disposent d'une option d'indicateur d'amorçage que vous pouvez utiliser pour tester vos charges de travail sur n'importe quel cluster pris en charge à l'aide de `containerd`. Pour plus d'informations, consultez Testez la migration Docker de `containerd`.

Nous continuerons à publier des AMI pour les versions existantes de Kubernetes jusqu'à la date de fin de support. Pour plus d'informations, consultez Calendrier de sortie de la version Kubernetes Amazon EKS. Si vous avez besoin de plus de temps pour tester vos charges de travail sur `containerd`, utilisez une version prise en charge antérieure à la version `1.24`. Mais si vous souhaitez mettre à niveau les AMI Amazon EKS officielles vers la version `1.24` ou ultérieure, vérifiez que vos charges de travail s'exécutent sur `containerd`.

L'environnement d'exécution `containerd` est plus fiables en termes de performances et de sécurité. `containerd` correspond à l'environnement d'exécution normalisé sur Amazon EKS. Fargate et Bottlerocket n'utilisent déjà que `containerd`. `containerd` réduit le nombre de versions d'AMI Amazon EKS nécessaires pour corriger les CVE (Vulnérabilités et expositions courantes) `Dockershim`. Comme `Dockershim` utilise déjà `containerd` en interne, vous n'aurez peut-être pas besoin d'apporter de modifications. Toutefois, dans certaines situations, des changements peuvent s'avérer nécessaires :

- Vous devrez apporter des modifications aux applications qui montent le socket Docker. Par exemple, les images de conteneurs créées à l'aide d'un conteneur seront affectées. De nombreux outils de surveillance montent également le socket Docker. Vous devrez peut-être attendre des mises à jour ou redéployer les charges de travail pour la surveillance de l'exécution.

- Vous devrez peut-être apporter des modifications pour les applications qui dépendent de paramètres Docker spécifiques. Par exemple, le protocole HTTPS_PROXY n'est plus pris en charge. Vous devez mettre à jour les applications qui utilisent ce protocole. Pour plus d'informations, consultez dockerd dans la documentation Docker.
- Si vous utilisez l'assistance des informations d'identification Amazon ECR pour extraire des images, vous devez basculer vers le fournisseur d'informations d'identification d'images kubelet. Pour plus d'informations, consultez Configurer un fournisseur d'informations d'identification d'image kubelet dans la documentation Kubernetes.
- Comme Amazon EKS 1.24 ne prend plus en charge Docker, certains indicateurs que le script d'amorçage Amazon EKS prenait auparavant en charge ne le sont plus. Avant de passer à Amazon EKS 1.24 ou version ultérieure, vous devez supprimer toute référence aux indicateurs qui ne sont plus pris en charge :
  - `--container-runtime dockerd` (`containerd` est la seule valeur prise en charge)
  - `--enable-docker-bridge`
  - `--docker-config-json`
- Si vous avez déjà configuré Fluentd pour Container Insights, vous devez effectuer la migration de Fluentd vers Fluent Bit avant de passer à `containerd`. Les analyseurs Fluentd sont configurés pour analyser uniquement les messages du journal au format JSON. Contrairement à `dockerd`, l'exécution du conteneur `containerd` a des messages de journal qui ne sont pas au format JSON. Si vous ne migrez pas vers Fluent Bit, certains des analyseurs Fluentd's configurés généreront un grand nombre d'erreurs à l'intérieur du conteneur Fluentd. Pour plus d'informations sur la migration, voir Configurer en Fluent Bit tant que DaemonSet pour envoyer des CloudWatch journaux vers Logs.
- Si vous utilisez une AMI personnalisée et que vous effectuez une mise à niveau vers Amazon EKS 1.24, vous devez vous assurer que le transfert IP est activé pour vos nœuds de travail. Ce paramètre n'était pas nécessaire avec Docker mais est requis pour `containerd`. Il est nécessaire pour résoudre les problèmes de connectivité des réseaux de Pod-à-Pod, de Pod-à-externe, ou de Pod-à-apiserver.

  Pour vérifier ce paramètre sur un nœud de travail, exécutez l'une des commandes suivantes :

  - `sysctl net.ipv4.ip_forward`
  - `cat /proc/sys/net/ipv4/ip_forward`

  Si le résultat est `0`, exécutez l'une des commandes suivantes pour activer la variable du noyau `net.ipv4.ip_forward` :

- `sysctl -w net.ipv4.ip_forward=1`
- `echo 1 > /proc/sys/net/ipv4/ip_forward`

Pour l'activation du paramètre sur les AMI Amazon EKS au cours de l'exécution `containerd`, consultez `install-worker.sh` sur GitHub.

AMI Amazon Linux optimisées pour Amazon EKS

L'AMI Amazon Linux optimisée pour Amazon EKS repose sur Amazon Linux 2 (AL2) et Amazon Linux 2023 (AL2023). Elle est configurée pour servir d'image de base pour les nœuds Amazon EKS. L'AMI est configurée pour fonctionner avec Amazon EKS et elle comprend les composants suivants :

- `kubelet`
- AWS Authentificateur IAM
- Docker (Version Amazon EKS `1.23` et version antérieure)
- `containerd`

> **Note**
>
> - Vous pouvez suivre les événements liés à la sécurité ou à la confidentialité d'AL2 dans le centre de sécurité Amazon Linux ou vous abonner au flux RSS associé. Les événements de sécurité et de confidentialité incluent une présentation du problème, les packages concernés et la manière de mettre à jour vos instances pour résoudre le problème.
> - Avant de déployer une AMI accélérée ou Arm, consultez les informations se trouvant dans AMI Amazon Linux accélérée optimisée pour Amazon EKS et AMI Amazon Linux Arm optimisées pour Amazon EKS.
> - Pour Kubernetes la version`1.23`, vous pouvez utiliser un indicateur bootstrap facultatif pour tester la migration de Docker vers`containerd`. Pour plus d'informations, consultez Testez la migration Docker de `containerd`.
> - À partir de Kubernetes version `1.25`, vous ne pourrez plus utiliser les instances `P2` Amazon EC2 avec les instances AMI Amazon Linux accélérées et optimisées pour Amazon EKS prêtes à l'emploi. Ces AMI pour Kubernetes `1.25` ou les versions ultérieures prendront en charge les pilotes de la série `NVIDIA 525` ou ultérieurs, qui sont incompatibles avec les instances `P2`. Toutefois, les pilotes de la série `NVIDIA 525` ou

ultérieurs sont compatibles avec les instances P3, P4 et P5, ce qui vous permet d'utiliser ces instances avec les AMI pour Kubernetes 1.25 ou une version ultérieure. Avant de mettre à niveau vos clusters Amazon EKS vers la version 1.25, migrez toutes les instances P2 vers les instances P3, P4 et P5. Vous devez également mettre à jour vos applications de manière proactive pour qu'elles fonctionnent avec les pilotes de la série NVIDIA 525 ou d'une série ultérieure. Nous prévoyons de rétroporter les nouvelles NVIDIA 525 séries ou les pilotes ultérieurs vers des Kubernetes versions ultérieures 1.23 et 1.24 ce, fin janvier 2024.

- Tout groupe de nœuds gérés nouvellement créé dans des clusters utilisant la version 1.30 ou une version plus récente utilisera automatiquement par défaut AL2023 comme système d'exploitation du nœud. Auparavant, les nouveaux groupes de nœuds étaient définis par défaut sur AL2. Vous pouvez continuer à utiliser AL2 en le choisissant comme type d'AMI lors de la création d'un nouveau groupe de nœuds.
- Support pour AL2 expirera le 30 juin 2025. Pour plus d'informations, consultez FAQ sur Amazon Linux 2.

Mise à niveau de AL2 à AL2023

L'AMI optimisée pour Amazon EKS est disponible en deux familles basées sur AL2 et AL2023. AL2023 est un nouveau système d'exploitation basé sur Linux conçu pour fournir un environnement sécurisé, stable et performant pour vos applications cloud. Il s'agit de la nouvelle génération d'Amazon Linux d'Amazon Web Services. Elle est disponible dans toutes les versions prises en charge d'Amazon EKS, y compris les versions 1.23 et 1.24 dans le cadre d'un support étendu. Les AMI accélérées Amazon EKS basées sur AL2023 seront disponibles ultérieurement. Si vous avez des charges de travail accélérées, vous devez continuer à utiliser l'AMI accélérée AL2 ou Bottlerocket.

AL2023 offre plusieurs améliorations par rapport à AL2. Pour une comparaison complète, consultez la section Comparaison entre AL2 et Amazon Linux 2023 dans le guide de l'utilisateur Amazon Linux 2023. Plusieurs packages ont été ajoutés, mis à niveau et supprimés d'AL2. Il est fortement recommandé de tester vos applications avec AL2023 avant de procéder à la mise à niveau. Pour obtenir la liste de toutes les modifications apportées aux packages dans AL2023, consultez la section Modifications apportées aux packages dans Amazon Linux 2023 dans les notes de mise à jour d'Amazon Linux 2023.

Outre ces modifications, vous devez être conscient de ce qui suit :

- AL2023 introduit un nouveau processus d'initialisation des nœuds `nodeadm` qui utilise un schéma de configuration YAML. Si vous utilisez des groupes de nœuds autogérés ou une AMI avec un modèle de lancement, vous devez désormais fournir des métadonnées de cluster supplémentaires de manière explicite lors de la création d'un nouveau groupe de nœuds. Voici un exemple des paramètres minimaux requis, où `apiServerEndpointcertificateAuthority`, et le service `cidr` sont désormais requis :

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

  Dans AL2, les métadonnées de ces paramètres ont été découvertes à partir de l'appel d'`DescribeCluster`API Amazon EKS. Avec AL2023, ce comportement a changé car l'appel d'API supplémentaire risque d'être limité lors de la mise à l'échelle de nœuds à grande échelle. Cette modification ne vous concerne pas si vous utilisez des groupes de nœuds gérés sans modèle de lancement ou si vous utilisezKarpenter. Pour plus d'informations sur les services `certificateAuthority` et les service`cidr`, consultez `DescribeCluster` le manuel Amazon EKS API Reference.

- Dockern'est pas pris en charge dans AL2023 pour toutes les versions d'Amazon EKS prises en charge. Support pour Docker a pris fin et a été supprimé avec la version Amazon EKS `1.24` ou supérieure dans AL2. Pour plus d'informations sur la dépréciation, consultez Amazon EKS a mis fin au support de. `Dockershim`
- La version CNI d'Amazon VPC `1.16.2` ou supérieure est requise pour AL2023.
- AL2023 nécessite `IMDSv2` par défaut. `IMDSv2`présente plusieurs avantages qui contribuent à améliorer la posture de sécurité. Il utilise une méthode d'authentification orientée session qui nécessite la création d'un jeton secret dans une simple requête HTTP PUT pour démarrer la session. Le jeton d'une session peut être valide entre 1 seconde et 6 heures. Pour plus d'informations sur la manière de passer de `IMDSv1` à`IMDSv2`, consultez Transition vers l'utilisation du service de métadonnées d'instance version 2 et Profitez de tous les avantages d'IMDSv2 et désactivez IMDSv1 dans votre infrastructure. AWS Si vous souhaitez l'utiliser`IMDSv1`, vous pouvez

toujours le faire en remplaçant manuellement les paramètres à l'aide des propriétés de lancement de l'option de métadonnées de l'instance.

> **Note**
>
> En effet`IMDSv2`, le nombre de sauts par défaut pour les groupes de nœuds gérés est défini sur 1. Cela signifie que les conteneurs n'auront pas accès aux informations d'identification du nœud via IMDS. Si vous avez besoin d'un accès par conteneur aux informations d'identification du nœud, vous pouvez toujours le faire en les remplaçant manuellement `HttpPutResponseHopLimit` dans un modèle de lancement Amazon EC2 personnalisé, en le portant à 2. Vous pouvez également utiliser Amazon EKS Pod Identity pour fournir des informations d'identification au lieu de`IMDSv2`.

- L'AL2023 propose la prochaine génération de hiérarchie unifiée des groupes de contrôle (`cgroupv2`). `cgroupv2`est utilisé pour implémenter un environnement d'exécution de conteneur, et par`systemd`. Bien que l'AL2023 inclue toujours du code permettant au système de fonctionner en utilisant`cgroupv1`, cette configuration n'est ni recommandée ni prise en charge. Cette configuration sera complètement supprimée dans une future version majeure d'Amazon Linux.
- `eksctl`une version `0.176.0` ou supérieure est requise pour `eksctl` prendre en charge AL2023.

Pour les groupes de nœuds gérés existants, vous pouvez effectuer une mise à niveau sur place ou une mise à niveau bleu/vert selon la manière dont vous utilisez un modèle de lancement :

- Si vous utilisez une AMI personnalisée avec un groupe de nœuds gérés, vous pouvez effectuer une mise à niveau sur place en échangeant l'ID de l'AMI dans le modèle de lancement. Vous devez d'abord vous assurer que vos applications et toutes les données utilisateur sont transférées vers AL2023 avant de mettre en œuvre cette stratégie de mise à niveau.
- Si vous utilisez des groupes de nœuds gérés avec le modèle de lancement standard ou avec un modèle de lancement personnalisé qui ne précise pas l'ID de l'AMI, vous devez effectuer la mise à niveau en utilisant une stratégie bleu/vert. Une mise à niveau bleu/vert est généralement plus complexe et implique la création d'un tout nouveau groupe de nœuds dans lequel vous devez spécifier AL2023 comme type d'AMI. Le nouveau groupe de nœuds devra ensuite être configuré avec soin pour garantir que toutes les données personnalisées du groupe de nœuds AL2 sont compatibles avec le nouveau système d'exploitation. Une fois que le nouveau groupe de nœuds a été testé et validé avec vos applications, il Pods peut être migré de l'ancien groupe de nœuds

vers le nouveau groupe de nœuds. Une fois la migration terminée, vous pouvez supprimer l'ancien groupe de nœuds.

Si vous utilisez Karpenter et souhaitez utiliser AL2023, vous devez modifier le `EC2NodeClass` `amiFamily` champ avec AL2023. Par défaut, Drift est activé dansKarpenter. Cela signifie qu'une fois le `amiFamily` champ modifié, vos nœuds de travail Karpenter seront automatiquement mis à jour avec la dernière AMI lorsqu'elle sera disponible.

AMI Amazon Linux accélérée optimisée pour Amazon EKS

> **Note**
>
> Les AMI accélérées Amazon EKS basées sur AL2023 seront disponibles ultérieurement. Si vous avez des charges de travail accélérées, vous devez continuer à utiliser l'AMI accélérée AL2 ou. Bottlerocket

L'AMI Amazon Linux accéléré optimisé pour Amazon EKS est construit sur l'AMI Amazon Linux standard optimisé pour Amazon EKS. Elle est configurée pour servir d'image facultative aux nœuds Amazon EKS afin de prendre en charge les charges de travail basées sur le GPU, Inferentia et Trainium.

Outre la configuration de l'AMI standard optimisée pour Amazon EKS, l'AMI accélérée inclut les éléments suivants :

- Pilotes NVIDIA
- `nvidia-container-runtime`
- AWS Neuronchauffeur

Pour obtenir la liste des derniers composants inclus dans l'AMI accélérée, consultez les `amazon-eks-ami` versions duGitHub.

> **Note**
>
> - L'AMI accélérée optimisée pour Amazon EKS ne prend en charge que les types d'instance basés sur GPU et Inferentia. Assurez-vous de spécifier ces types d'instances dans votre

AWS CloudFormation modèle de nœud. En utilisant l'AMI accélérée optimisée pour Amazon EKS vous acceptez le Contrat de licence de l'utilisateur final NVIDIA (CLUF).

- L'AMI accélérée optimisée pour Amazon EKS était auparavant appelée AMI optimisée pour Amazon EKS avec prise en charge du GPU.
- Les versions précédentes de l'AMI accélérée optimisée pour Amazon EKS ont installé le référentiel `nvidia-docker`. Le référentiel n'est plus inclus dans l'AMI Amazon EKS version `v20200529` et ultérieure.

Pour activer les charges de travail basées AWS sur Neuron (accélérateur ML)

Pour en savoir plus sur les charges de travail de formation et d'inférence utilisées dans Neuron Amazon EKS, consultez les références suivantes :

- Conteneurs - Kubernetes - Commencer par la documentation AWS Neuron
- Formation dans AWS Neuron EKS Samples sur GitHub
- Inférence de machine learning à l'aide de AWS Inferentia

Pour activer des applications basées sur GPU

La procédure suivante décrit comment exécuter une application sur une instance GPU avec l'AMI accélérée optimisée pour Amazon EKS.

1. Une fois vos nœuds GPU ajoutés à votre cluster, vous devez mettre en œuvre le plugin de périphérique NVIDIA pour Kubernetes en tant que DaemonSet sur votre cluster. Remplacez *vX.X.X* par la version NVIDIA/k8s-device-plugin souhaitée avant d'exécuter la commande suivante.

   ```
   kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
   nvidia-device-plugin.yml
   ```

2. Vous pouvez vérifier que vos nœuds ont des GPU répartis avec la commande suivante.

   ```
   kubectl get nodes "-o=custom-
   columns=NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"
   ```

Pour déployer un Pod pour tester que vos nœuds GPU sont configurés correctement

1. Créez un fichier nommé `nvidia-smi.yaml` avec les contenus suivants. Remplacez *`tag`* par la balise souhaitée pour `nvidia/cuda`. Ce manifeste lance un conteneur NVIDIA CUDA qui exécute `nvidia-smi` sur un nœud.

```
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-smi
spec:
  restartPolicy: OnFailure
  containers:
  - name: nvidia-smi
    image: nvidia/cuda:tag
    args:
    - "nvidia-smi"
    resources:
      limits:
        nvidia.com/gpu: 1
```

2. Appliquez le manifeste ci-dessus avec la commande suivante.

```
kubectl apply -f nvidia-smi.yaml
```

3. Une fois que le Pod n'est plus en cours d'exécution, affichez ses journaux à l'aide de la commande suivante.

```
kubectl logs nvidia-smi
```

L'exemple qui suit illustre un résultat.

```
Mon Aug  6 20:23:31 20XX
+-----------------------------------------------------------------------------+
| NVIDIA-SMI XXX.XX                 Driver Version: XXX.XX                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:00:1C.0 Off |                    0 |
| N/A   46C    P0    47W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
```

```
+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

AMI Amazon Linux Arm optimisées pour Amazon EKS

Les instances Arm apportent des économies significatives en termes de coût et sont bien adaptées aux applications dimensionnables et basées sur Arm, telles que serveurs web, microservices conteneurisés, flottes de cache et magasins de données distribuées. Lorsque vous ajoutez des nœuds Arm à votre cluster, passez en revue les considérations suivantes.

Considérations

- Si votre cluster a été déployé avant le 17 août 2020, vous devez effectuer une mise à niveau unique des manifestes des modules complémentaires critiques du cluster. Ceci afin que Kubernetes puisse extraire l'image correcte pour chaque architecture matérielle utilisée dans votre cluster. Pour plus d'informations sur la mise à jour des modules complémentaires de clusters, consultez Mise à jour de la version Kubernetes de votre cluster Amazon EKS. Si vous avez déployé votre cluster le 17 août 2020 ou après cette date, vos modules complémentaires CoreDNS, `kube-proxy`, et Amazon VPC CNI plugin for Kubernetes sont déjà compatibles avec plusieurs architectures.
- Les applications déployées sur les nœuds Arm doivent être compilées pour Arm.
- Si vous avez déployé des DaemonSets dans un cluster existant, ou si vous souhaitez les déployer dans un nouveau cluster dans lequel vous souhaitez également déployer des nœuds Arm, vérifiez que votre DaemonSet peut s'exécuter sur toutes les architectures matérielles de votre cluster.
- Vous pouvez exécuter des groupes de nœuds Arm et des groupes de nœuds x86 dans le même cluster. Si vous procédez de la sorte, envisagez de déployer des images de conteneur multi-architecture dans un référentiel de conteneurs tel qu'Amazon Elastic Container Registry, puis d'ajouter des sélecteurs de nœuds à vos manifestes afin que Kubernetes connaisse l'architecture matérielle dans laquelle un Pod peut être déployé. Pour de plus amples informations, consultez Transmission d'une image multi-architecture dans le Guide de l'utilisateur Amazon ECR et l'article de blog Présentation d'images de conteneurs multi-architectures pour Amazon ECR.

Testez la migration Docker de `containerd`

Amazon EKS a mis fin à la prise en charge de Docker à partir du lancement de la version `1.24` de Kubernetes. Pour plus d'informations, consultez Amazon EKS a mis fin à la prise en charge de `Dockershim`.

Pour ce qui est de la Kubernetes version`1.23`, vous pouvez utiliser un indicateur bootstrap facultatif pour activer l'`containerd`exécution des AMI AL2 optimisées pour Amazon EKS. Cette fonctionnalité vous offre une voie claire pour migrer vers `containerd` lors de la mise à jour vers la version `1.24` ou ultérieure. Amazon EKS a mis fin à la prise en charge de Docker à partir du lancement de la version `1.24` de Kubernetes. L'environnement d'exécution `containerd` est largement adopté par la communauté Kubernetes et est un projet gradué de la CNCF. Vous pouvez le tester en ajoutant un groupe de nœuds à un cluster nouveau ou existant.

Vous pouvez activer l'indicateur d'amorçage en créant l'un des types de groupes de nœuds suivants.

Autogéré

Créez le groupe de nœuds à l'aide des instructions contenues dans Lancement de nœuds Amazon Linux autogérés. Spécifiez une AMI optimisée pour Amazon EKS et le texte suivant pour le paramètre `BootstrapArguments`.

```
--container-runtime containerd
```

Gérées

Si vous utilisez `eksctl`, créez un fichier nommé *`my-nodegroup`*`.yaml` avec le contenu suivant. Remplacez chaque *`example value`* par vos propres valeurs. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Pour récupérer un ID d'AMI optimisée pour `ami-`*`1234567890abcdef0`*, veuillez consulter la rubrique Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
  version: 1.23
managedNodeGroups:
  - name: my-nodegroup
```

```
    ami: ami-1234567890abcdef0
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

> **Note**
>
> Si vous lancez de nombreux nœuds simultanément, vous pouvez également spécifier des valeurs pour le `--apiserver-endpoint`, `--b64-cluster-ca`, et les arguments bootstrap d'amorçage `--dns-cluster-ip` pour éviter les erreurs. Pour plus d'informations, consultez Spécification d'une AMI.

Exécutez les commandes suivantes pour créer le groupe de nœuds.

```
eksctl create nodegroup -f my-nodegroup.yaml
```

Si vous préférez utiliser un autre outil pour créer votre groupe de nœuds gérés, vous devez déployer le groupe de nœuds à l'aide d'un modèle de lancement. Dans votre modèle de lancement, spécifiez un ID d'AMI optimisée pour Amazon EKS, puis déployez le groupe de nœuds avec un modèle de lancement et fournissez les données utilisateur suivantes. Ces données utilisateur transmettent des arguments dans le fichier `bootstrap.sh`. Pour plus d'informations sur le fichier d'amorçage, consultez bootstrap.sh sur GitHub.

```
/etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

En savoir plus

Pour plus d'informations sur l'utilisation d'AMI Amazon Linux optimisées pour Amazon EKS, veuillez consulter les sections suivantes :

- Pour utiliser Amazon Linux avec des groupes de nœuds gérés, veuillez consulter la rubrique Groupes de nœuds gérés.
- Pour lancer des nœuds Amazon Linux autogérés, veuillez consulter la rubrique Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS.
- Pour plus d'informations sur la version, consultez Versions d'AMI Amazon Linux optimisées pour Amazon EKS.

- Pour récupérer les derniers ID des AMI Amazon Linux optimisées pour Amazon EKS, veuillez consulter la rubrique Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS.
- Pour consulter les scripts open source qui sont utilisés pour générer l'AMI optimisée pour Amazon EKS, veuillez consulter la rubrique Script de création d'AMI Amazon Linux optimisées pour Amazon EKS.

Versions d'AMI Amazon Linux optimisées pour Amazon EKS

Les AMI optimisées par Amazon EKS Linux sont versionnées par la version Kubernetes et la date de sortie de l'AMI dans le format suivant :

```
k8s_major_version.k8s_minor_version.k8s_patch_version-release_date
```

Chaque version d'AMI inclut différentes versions de `kubelet`, de Docker, du noyau Linux et de `containerd`. L'AMI accélérée inclut également différentes versions du pilote NVIDIA. Vous pouvez trouver ces informations de version dans le Journal des modifications sur GitHub.

Récupération des ID d'AMI Amazon Linux optimisées pour Amazon EKS

Vous pouvez récupérer par programmation l'identifiant Amazon Machine Image (AMI) pour les AMI optimisées Amazon EKS en interrogeant l'API AWS Systems Manager Parameter Store. Ce paramètre vous évite de devoir rechercher manuellement les ID d'AMI optimisées pour Amazon EKS. Pour plus d'informations sur l'API Systems Manager Parameter Store, consultez GetParameter.

Pour récupérer un ID d'AMI pour les AMI optimisées pour Amazon EKS à l'aide du AWS CLI

1. Déterminez la région dans laquelle votre instance de nœud sera déployée, par exemple`us-west-2`.
2. Déterminez le type d'AMI dont vous avez besoin. Pour plus d'informations sur les types d'instances Amazon EC2, consultez la section Types d'instances.

   - `amazon-linux-2`est destiné aux instances x86 basées sur Amazon Linux 2 (AL2).
   - `amazon-linux-2-arm64`est destiné aux instances ARM AL2, telles que les instances basées sur AWS Graviton.
   - `amazon-linux-2-gpu`est destiné aux instances accélérées par GPU AL2.
   - `amazon-linux-2023/x86_64/standard`est destiné aux instances x86 basées sur Amazon Linux 2023 (AL2023).

- `amazon-linux-2023/arm64/standard`est destiné aux instances ARM AL2023.

3. Déterminez la Kubernetes version du cluster auquel votre nœud sera attaché, par exemple 1.30.
4. Exécutez la AWS CLI commande suivante pour récupérer l'ID d'AMI approprié. Remplacez Région AWS la Kubernetes version et la plate-forme le cas échéant. Vous devez être connecté à l' AWS CLI aide d'un principal `ssm:GetParameter` IAM autorisé à récupérer les métadonnées de l'AMI optimisées pour Amazon EKS.

```
aws ssm get-parameter --name /aws/service/eks/optimized-ami/1.30/amazon-linux-2/
recommended/image_id \
                --region region-code --query "Parameter.Value" --output text
```

L'exemple qui suit illustre un résultat.

```
ami-1234567890abcdef0
```

Script de création d'AMI Amazon Linux optimisées pour Amazon EKS

Amazon Elastic Kubernetes Service (Amazon EKS) comporte des scripts open source qui sont utilisés pour créer l'AMI optimisée pour Amazon EKS. Ces scripts de génération sont disponibles sur GitHub.

L'AMI Amazon Linux optimisée pour Amazon EKS est basée sur Amazon Linux 2 (AL2) et Amazon Linux 2023 (AL2023), spécifiquement pour être utilisée comme nœud dans les clusters Amazon EKS. Vous pouvez utiliser ce référentiel pour voir les détails de la manière dont l'équipe Amazon EKS configure `kubelet` l'authentificateur AWS IAM pour Kubernetes et crée votre propre AMI basée sur Amazon Linux à partir de zéro. Docker

Le référentiel de scripts de génération inclut un modèle de HashiCorppacker et des scripts de génération pour générer une AMI. Ces scripts sont une source fiable pour les générations d'AMI optimisées pour Amazon EKS. Vous pouvez donc suivre le référentiel GitHub pour surveiller les modifications apportées à notre AMI. Par exemple, vous pouvez souhaiter que votre propre AMI utilise la même version de Docker que celle utilisée par l'équipe Amazon EKS pour l'AMI officielle.

Le GitHub référentiel contient également le script bootstrap spécialisé et le script nodeadm qui s'exécutent au démarrage pour configurer les données de certificat de votre instance, le point de terminaison du plan de contrôle, le nom du cluster, etc.

En outre, le GitHub référentiel contient nos AWS CloudFormation modèles de nœuds Amazon EKS. Ces modèles facilitent le lancement d'une instance exécutant l'AMI optimisée pour Amazon EKS et son enregistrement avec un cluster.

Pour plus d'informations, consultez les référentiels sur GitHub à l'adresse https://github.com/awslabs/amazon-eks-ami.

L'AL2 optimisé pour Amazon EKS contient un indicateur de démarrage facultatif pour activer le `containerd` runtime.

Configuration VT1 pour votre AMI Amazon Linux personnalisée

Les AMI Amazon Linux personnalisées dans Amazon EKS peuvent prendre en charge la famille d'instances de transcodage vidéo VT1 pour Amazon Linux 2 (AL2), Ubuntu 18 et 20. Ubuntu VT1prend en charge les cartes de transcodage multimédia Xilinx U30 avec les codecs H.264/AVC et H.265/HEVC accélérés. Pour bénéficier de ces instances accélérées, vous devez suivre les étapes suivantes :

1. Créez et lancez une AMI de base à partir d'AL2, Ubuntu 18 ou Ubuntu 20.
2. Après le lancement de l'AMI basée, installez le Pilote XRT et l'exécution sur le nœud.
3. Création d'un cluster Amazon EKS.
4. Installez le plugin FPGA Kubernetes sur votre cluster.

```
kubectl apply -f fpga-device-plugin.yml
```

Le plugin annonce désormais les appareils Xilinx U30 par nœud sur votre cluster Amazon EKS. Vous pouvez utiliser l'image FFMPEG docker pour exécuter des exemples de charges de travail de transcodage vidéo sur votre cluster Amazon EKS.

Configuration DL1 pour votre AMI Amazon Linux 2 personnalisée

Les AMI Amazon Linux 2 (AL2) personnalisées dans Amazon EKS peuvent prendre en charge les charges de travail d'apprentissage profond à grande échelle grâce à une configuration supplémentaire et Kubernetes à des modules complémentaires. Ce document décrit les composants nécessaires pour configurer une solution Kubernetes générique pour une configuration sur site ou comme base de référence dans une configuration cloud plus importante. Pour prendre en charge cette fonction, vous devrez effectuer les étapes suivantes dans votre environnement personnalisé :

- SynapaseAI® Softwarepilotes chargés sur le système — Ils sont inclus dans les AMI disponibles sur Github.
- Le plug-in pour Habana appareils — Un plug-in DaemonSet qui vous permet d'activer automatiquement l'enregistrement des Habana appareils dans votre Kubernetes cluster et de suivre l'état de santé des appareils.
- Helm 3.x
- Les Charts de Helm pour installer l'opérateur MPI.
- L'opérateur MPI

1. Créez et lancez une AMI de base à partir d'AL2, Ubuntu 18 ou Ubuntu 20.
2. Suivez ces instructions pour configurer l'environnement pourDL1.

AMI Bottlerocket optimisées pour Amazon EKS

Bottlerocket est une distribution open source Linux sponsorisée et soutenue par AWS. Bottlerocket est spécialement conçu pour héberger les charges de travail des conteneurs. Avec Bottlerocket, vous pouvez ainsi améliorer la disponibilité des déploiements conteneurisés et réduire les coûts opérationnels en automatisant les mises à jour de votre infrastructure de conteneurs. Bottlerocket inclut uniquement les logiciels essentiels pour exécuter des conteneurs, ce qui améliore l'utilisation des ressources, réduit les menaces de sécurité et réduit les frais généraux. L'AMI Bottlerocket inclut `containerd`, `kubelet` et l'authentificateur IAM AWS. Outre les groupes de nœuds gérés et les nœuds autogérés, Bottlerocket est également pris en charge par Karpenter.

Avantages

L'utilisation de Bottlerocket avec votre cluster Amazon EKS présente les bénéfices suivants :

- Une disponibilité accrue associée à des coûts opérationnels réduits et à une gestion moins complexe – Bottlerocket nécessite moins de ressources, délais de démarrage plus courts et est moins vulnérable aux menaces de sécurité que les autres distributions Linux. Une couverture plus petite de Bottlerocket's permet de réduire les coûts en utilisant moins de ressources de stockage, de calcul et de mise en réseau.
- Sécurité améliorée grâce aux mises à jour automatiques du système d'exploitation : les mises à jour de Bottlerocket sont appliquées en une seule unité et peuvent être annulées si nécessaire. Cela élimine le risque de mises à jour corrompues ou échouées qui peuvent laisser le système

dans un état inutilisable. Avec Bottlerocket, les mises à jour de sécurité peuvent être appliquées automatiquement dès qu'elles sont disponibles de manière à perturber le moins possible l'activité et peuvent être annulées en cas de défaillance.

- Premium Support – les versions AWS fournies sur Amazon EC2 Bottlerocket sont couvertes par les mêmes plans de support AWS Support qui couvrent également des services AWS tels qu'Amazon EC2, Amazon EKS et Amazon ECR.

Considérations

Tenez compte des points suivants lors de l'utilisation de Bottlerocket pour votre type d'AMI :

- Bottlerocket prend en charge les instances Amazon EC2 avec processeurs `x86_64` et `arm64`. Il n'est pas recommandé d'utiliser une AMI Bottlerocket avec les instances Amazon EC2 dotées d'une puce Inferentia.
- Actuellement, il n'existe aucun modèle AWS CloudFormation avec lequel vous pouvez déployer des nœuds Bottlerocket.
- Les images Bottlerocket n'incluent pas un serveur SSH ou un shell. Vous pouvez utiliser des méthodes d'accès hors bande pour autoriser SSH. Ces approches activent le conteneur d'administration et transmettent certaines étapes de configuration de l'action d'amorçage avec des données utilisateur. Pour de plus amples informations, veuillez consulter les sections suivantes dans la rubrique Système d'exploitation Bottlerocket sur GitHub :
  - Exploration
  - Conteneur d'administration
  - Paramètres Kubernetes
- Bottlerocket utilise différents types de conteneur :
  - Par défaut, est activé un conteneur de contrôle. Ce conteneur exécute l'agent AWS Systems Manager que vous pouvez utiliser pour exécuter des commandes ou démarrer des sessions shell sur les instances Bottlerocket d'Amazon EC2. Pour plus d'informations, consultez Configuration du gestionnaire de session dans le Guide de l'utilisateur AWS Systems Manager.
  - Un conteneur d'administration est activé si une clé SSH est fournie lors de la création du groupe de nœuds. Nous recommandons d'utiliser le conteneur d'administration uniquement pour les scénarios de développement et de test. Nous ne recommandons pas de l'utiliser pour les environnements de production. Pour plus d'informations, consultez Conteneur d'administration sur GitHub.

En savoir plus

Pour plus d'informations sur l'utilisation d'AMI Bottlerocket optimisées pour Amazon EKS, veuillez consulter les sections suivantes :

- Pour plus d'informations sur Bottlerocket, veuillez consulter la documentation et les versions sur GitHub.
- Pour utiliser Bottlerocket avec des groupes de nœuds gérés, veuillez consulter la rubrique Groupes de nœuds gérés.
- Pour lancer des nœuds Bottlerocket autogérés, veuillez consulter la rubrique Lancement de nœuds Bottlerocket autogérés.
- Pour récupérer les derniers ID des AMI Bottlerocket optimisées pour Amazon EKS, veuillez consulter la rubrique Récupération des ID d'AMI Bottlerocket optimisées pour Amazon EKS.
- Pour plus de détails sur la prise en charge en matière de conformité, veuillez consulter la rubrique Prise en charge de la conformité Bottlerocket.

Récupération des ID d'AMI Bottlerocket optimisées pour Amazon EKS

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour les AMI optimisées pour Amazon EKS en interrogeant l'API AWS Systems Manager Parameter Store. Grâce à ce paramètre, vous n'avez pas besoin de rechercher manuellement les ID d'AMI optimisées pour Amazon EKS. Pour plus d'informations sur l'API Systems Manager Parameter Store, consultez GetParameter. Le principal IAM que vous utilisez doit disposer de l'autorisation IAM `ssm:GetParameter` pour récupérer les métadonnées de l'AMI optimisée pour Amazon EKS.

Vous pouvez récupérer l'ID d'image de la dernière Bottlerocket AMI optimisée Amazon EKS recommandée à l'aide de la AWS CLI commande suivante en utilisant le sous-paramètre`image_id`. Remplacez *`1.30`* par une version prise en charge et *`region-code`* par une région prise en charge par Amazon EKS pour laquelle vous souhaitez obtenir l'ID d'AMI.

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-1.30/x86_64/latest/
image_id --region region-code --query "Parameter.Value" --output text
```

L'exemple qui suit illustre un résultat.

```
ami-1234567890abcdef0
```

Prise en charge de la conformité Bottlerocket

Bottlerocket est conforme aux recommandations définies par différentes organisations :

- Une évaluation CIS est défini pour Bottlerocket. Dans une configuration par défaut, l'image Bottlerocket possède la plupart des contrôles requis par le profil de configuration CIS de niveau 1. Vous pouvez implémenter les contrôles requis pour un profil de configuration CIS de niveau 2. Pour plus d'informations, veuillez consulter Validation de l'AMI Bottlerocket optimisée pour Amazon EKS par rapport à l'évaluation CIS sur le blog AWS.
- L'ensemble de fonctionnalités optimisé et la surface d'attaque réduite signifient que les instances Bottlerocket nécessitent moins de configuration pour répondre aux exigences de la norme PCI DSS. L'évaluation CIS pour Bottlerocket est une excellente ressource pour les conseils de renforcement et répond à vos exigences en matière de normes de configuration sécurisées conformément à l'exigence 2.2 de la norme PCI DSS. Vous pouvez également tirer parti de Fluent Bit pour répondre à vos exigences en matière de journalisation des audits au niveau du système d'exploitation conformément à la norme PCI DSS 10.2. AWS publie régulièrement de nouvelles instances Bottlerocket (corrigées) pour vous aider à répondre à l'xigence PCI DSS 6.2 (pour la version 3.2.1) et à l'exigence 6.3.3 (pour la version 4.0).
- Bottlerocket est une fonctionnalité éligible HIPAA dont l'utilisation est autorisée avec des charges de travail réglementées à la fois pour Amazon EC2 et Amazon EKS. Pour plus d'informations, consultez le livre blanc Architecting for HIPAA Security and Compliance on Amazon EKS.

AMI Ubuntu Linux optimisées pour Amazon EKS

Canonical a établi un partenariat avec Amazon EKS pour créer des AMI de nœuds que vous pouvez utiliser dans vos clusters.

Canonical fournit une image du système d'exploitation built-for-purpose Kubernetes Node. Cette Ubuntu image réduite est optimisée pour Amazon EKS et inclut le AWS noyau personnalisé développé conjointement avec AWS. Pour plus d'informations, consultez UbuntuAmazon Elastic Kubernetes Service (EKS) et. Lancement de nœuds Ubuntu autogérés Pour plus d'informations sur la prise en charge, consultez la section Logiciels tiers de Questions fréquentes sur AWS Premium Support.

AMI Windows optimisées pour Amazon EKS

Les AMI optimisées Windows pour Amazon EKS reposent sur Windows Server 2019 et Windows Server 2022. Elles sont configurées pour servir d'image de base pour les nœuds Amazon EKS. Par défaut, les AMI incluent les composants suivants :

- `kubelet`
- `kube-proxy`
- AWS Authentificateur IAM pour Kubernetes
- `csi-proxy`
- `containerd`

> **Note**
>
> Vous pouvez suivre les événements de sécurité ou de confidentialité pour Windows Server à l'aide du Guide de mise à jour de sécurité Microsoft.

Amazon EKS propose des AMI qui sont optimisées pour les conteneurs Windows dans les variantes suivantes :

- AMI Windows Server 2019 Core optimisée pour Amazon EKS
- AMI Windows Server 2019 Full optimisée pour Amazon EKS
- AMI Windows Server 2022 Core optimisée pour Amazon EKS
- AMI Windows Server 2022 Full optimisée pour Amazon EKS

> **Important**
>
> - L'AMI Windows Server 20H2 Core optimisée pour Amazon EKS est obsolète. Aucune nouvelle version de cette AMI ne sera publiée.
> - Pour garantir que vous disposez des dernières mises à jour de sécurité par défaut, Amazon EKS gère des Windows AMI optimisées au cours des 4 derniers mois. Chaque nouvelle AMI sera disponible pendant 4 mois à compter de la date de publication initiale. Après cette période, les anciennes AMI deviennent privées et ne sont plus accessibles. Nous

vous recommandons d'utiliser les dernières AMI afin d'éviter les failles de sécurité et de perdre l'accès aux anciennes AMI qui ont atteint la fin de leur durée de vie prise en charge. Bien que nous ne puissions pas garantir l'accès aux AMI qui ont été rendues privées, vous pouvez demander l'accès en déposant un ticket auprès de AWS Support.

Calendrier des versions

Le tableau suivant répertorie les dates de publication et de fin de prise en charge des versions Windows sur Amazon EKS. Si une date de fin est vide, c'est parce que la version est toujours prise en charge.

Version de Windows	Version d'Amazon EKS	Fin de support pour Amazon EKS
Windows Server 2022 Core	10/17/2022	
Windows Server 2022 Full	10/17/2022	
Windows Server 20H2 Core	8/12/2021	8/9/2022
Windows Server 2004 Core	8/19/2020	12/14/2021
Windows Server 2019 Core	10/7/2019	
Windows Server 2019 Full	10/7/2019	
Windows Server 1909 Core	10/7/2019	12/8/2020

Paramètres de configuration du script d'amorçage

Lorsque vous créez un nœud Windows, un script sur le nœud permet la configuration de différents paramètres. En fonction de votre configuration, ce script peut être trouvé sur le nœud à un emplacement similaire à :`C:\Program Files\Amazon\EKS\Start-EKSBootstrap.ps1`. Vous pouvez indiquer des valeurs de paramètres personnalisées en les spécifiant comme arguments dans le script d'amorçage. Par exemple, vous pouvez mettre à jour les données utilisateur dans le modèle de lancement. Pour plus d'informations, consultez Données utilisateur Amazon EC2.

Le script comprend les paramètres de ligne de commande suivants :

- `-EKSClusterName` : spécifie le nom du cluster Amazon EKS que ce composant master doit rejoindre.
- `-KubeletExtraArgs` : spécifie des arguments supplémentaires pour `kubelet` (facultatif).
- `-KubeProxyExtraArgs` : spécifie des arguments supplémentaires pour `kube-proxy` (facultatif).
- `-APIServerEndpoint` : spécifie le point de terminaison du serveur d'API de cluster Amazon EKS (facultatif). Uniquement valable lorsqu'il est utilisé avec `-Base64ClusterCA`. Évite d'appeler `Get-EKSCluster`.
- `-Base64ClusterCA` : spécifie le contenu de l'autorité de certification du cluster codée en base64 (facultatif). Uniquement valable lorsqu'il est utilisé avec `-APIServerEndpoint`. Évite d'appeler `Get-EKSCluster`.
- `-DNSClusterIP` : remplace l'adresse IP à utiliser pour les requêtes DNS au sein du cluster (facultatif). La valeur par défaut est `10.100.0.10` ou `172.20.0.10` en fonction de l'adresse IP de l'interface principale.
- `-ServiceCIDR` : remplace la plage d'adresses IP du service Kubernetes à partir de laquelle les services du cluster sont adressés. La valeur par défaut est `172.20.0.0/16` ou `10.100.0.0/16` en fonction de l'adresse IP de l'interface principale.
- `-ExcludedSnatCIDRs` : liste des CIDR `IPv4` à exclure de la traduction d'adresses réseau source (SNAT, Source Network Address Translation). Cela signifie que l'adresse IP privée du pod qui est adressable par le VPC ne serait pas traduite en adresse IP de l'adresse `IPv4` principale de l'instance ENI pour le trafic sortant. Par défaut, le CIDR `IPv4` du VPC pour le nœud Windows Amazon EKS est ajouté. La spécification des CIDR à ce paramètre exclut également les CIDR spécifiés. Pour plus d'informations, consultez SNAT pour Pods.

En plus des paramètres de la ligne de commande, vous pouvez également spécifier certains paramètres de la variable d'environnement. Lorsqu'un paramètre de ligne de commande est spécifié, il est prioritaire sur la variable d'environnement correspondante. La ou les variables d'environnement doivent être définies au niveau de la machine (ou du système), car le script d'amorçage ne lira que les variables au niveau de la machine.

Le script prend en compte les variables d'environnement suivantes :

- `SERVICE_IPV4_CIDR`— Reportez-vous au paramètre de la ligne de commande `ServiceCIDR` pour la définition.
- `EXCLUDED_SNAT_CIDRS`— Il doit s'agir d'une chaîne de caractères séparée par des virgules. Reportez-vous au paramètre de la ligne de commande `ExcludedSnatCIDRs` pour la définition.

Lancez des nœuds autogérés Windows Server 2022 avec **eksctl**

Vous pouvez utiliser le fichier **test-windows-2022.**yaml suivant comme référence pour exécuter Windows Server 2022 en tant que nœuds autogérés. Remplacez chaque *example value* par vos propres valeurs.

> **Note**
>
> Vous devez utiliser eksctl version 0.116.0 ou ultérieure pour exécuter des nœuds Windows Server 2022 autogérés.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: windows-2022-cluster
  region: region-code
  version: '1.30'

nodeGroups:
  - name: windows-ng
    instanceType: m5.2xlarge
    amiFamily: WindowsServer2022FullContainer
    volumeSize: 100
    minSize: 2
    maxSize: 3
  - name: linux-ng
    amiFamily: AmazonLinux2
    minSize: 2
    maxSize: 3
```

Les groupes de nœuds peuvent ensuite être créés à l'aide de la commande suivante.

```
eksctl create cluster -f test-windows-2022.yaml
```

Prise en charge de l'authentification gMSA

Les Pods Windows Amazon EKS autorisent différents types d'authentification de compte de service géré de groupe (gMSA).

- Amazon EKS prend en charge les identités de domaine Active Directory à des fins d'authentification. Pour plus d'informations sur gMSA joint à un domaine, veuillez consulter la rubrique Authentification Windows sur les podsWindows Amazon EKS sur le blog AWS .
- Amazon EKS propose un plug-in qui permet aux non-domain-joined Windows nœuds de récupérer des gMSA informations d'identification à l'aide d'une identité utilisateur portable. Pour plus d'informations sur gMSA sans domaine, veuillez consulter la rubrique Authentification Windows sans domaine pour les podsWindows Amazon EKS sur le blog AWS .

Images de conteneur en cache

Certaines images de conteneur des AMI optimisées pour Windows d'Amazon EKS sont mises en cache pour l'`containerd`exécution. Les images de conteneur sont mises en cache lors de la génération d'AMI personnalisées à l'aide de composants de génération gérés par Amazon. Pour plus d'informations, consultez Utilisation du composant de génération géré par Amazon.

Les images de conteneur mises en cache suivantes sont destinées à l'exécution `containerd` :

- `amazonaws.com/eks/pause-windows`
- `mcr.microsoft.com/windows/nanoserver`
- `mcr.microsoft.com/windows/servercore`

En savoir plus

Pour plus d'informations sur l'utilisation d'AMI Windows optimisées pour Amazon EKS, veuillez consulter les sections suivantes :

- Pour utiliser Windows avec des groupes de nœuds gérés, veuillez consulter la rubrique Groupes de nœuds gérés.
- Pour lancer des nœuds Windows autogérés, veuillez consulter la rubrique Lancement de nœuds Windows autogérés.
- Pour plus d'informations sur la version, consultez Versions d'AMI Windows optimisée pour Amazon EKS.
- Pour récupérer les derniers ID des AMI Windows optimisées pour Amazon EKS, veuillez consulter la rubrique Récupération des ID d'AMI Windows optimisées pour Amazon EKS.

- Pour utiliser Amazon EC2 Image Builder pour créer des AMI Windows optimisées pour Amazon EKS, veuillez consulter la rubrique Création d'AMI Windows optimisées pour Amazon EKS personnalisées.
- Pour connaître les meilleures pratiques, consultez la section Gestion optimisée des Windows AMI Amazon EKS dans le guide des meilleures pratiques d'EKS.

Versions d'AMI Windows optimisée pour Amazon EKS

> **Important**
>
> Support étendu pour les Windows AMI optimisées Amazon EKS publiées par AWS n'est pas disponible pour les Kubernetes versions `1.23` mais est disponible pour les Kubernetes versions `1.24` et supérieures.

Cette rubrique répertorie les versions des Windows AMI optimisées pour Amazon EKS et leurs versions correspondantes de `kubelet`, `containerd`, et `csi-proxy`.

Les métadonnées d'AMI optimisées pour Amazon EKS, notamment l'ID d'AMI, peuvent être extraites par programmation pour chaque variante. Pour plus d'informations, consultez Récupération des ID d'AMI Windows optimisées pour Amazon EKS.

Les AMI sont versionnées par la version Kubernetes et la date de sortie de l'AMI dans le format suivant :

```
k8s_major_version.k8s_minor_version-release_date
```

> **Note**
>
> Les groupes de nœuds gérés par Amazon EKS prennent en charge les versions de novembre 2022 et ultérieures des Windows AMI.

AMI Windows Server 2022 Core optimisée pour Amazon EKS

Le tableau ci-dessous répertorie les versions actuelles et précédentes de l'AMI Windows Server 2022 Core optimisée pour Amazon EKS.

Kubernetes version 1,30

KubernetesVersion**1.30**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

KubernetesVersion**1.29**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Mise à niveau `containerd` vers `1.7.11`. Mise à niveau `kubelet` vers `1.29.3`.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Mise à niveau `containerd` vers `1.6.28`. CNI reconstruit et `csi-proxy` utilisation `golang1.22.1`.
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Correction d'un bug à cause duquel l'image de pause était supprimée de manière incorrect

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
				e par le processus de collecte `kubelet` des déchets.
`1.29-2024.01.11`	`1.29.0`	`1.6.18`	`1.1.2`	WindowsMise à jour autonome KB5034439 exclue sur Windows les AMI principales du serveur 2022. La base de connaissances s'applique uniquement aux Windows installations avec une WinRE partition séparée, qui ne sont incluses dans aucune de nos Windows AMI optimisées Amazon EKS.

Kubernetes version 1,28

KubernetesVersion**`1.28`**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
`1.28-2024.05.14`	`1.28.8`	`1.6.28`	`1.1.2`	Mise à niveau `containerd` vers `1.6.28`. Mise à niveau `kubelet` vers `1.28.8`.
`1.28-2024.04.09`	`1.28.5`	`1.6.25`	`1.1.2`	Mise à niveau `containerd` vers `1.6.25`. CNI reconstruit et `csi-proxy` utilisation `golang1.22.1`.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.11	1.28.5	1.6.18	1.1.2	WindowsMise à jour autonome KB5034439 exclue sur Windows les AMI principales du serveur 2022. La base de connaissances s'applique uniquement aux Windows installations avec une WinRE partition séparée, qui ne sont incluses dans aucune de nos Windows AMI optimisées Amazon EKS.
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1,27

KubernetesVersion**1.27**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	Mise à niveau `containerd` vers `1.6.28`. Mise à niveau `kubelet` vers `1.27.12`.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Mise à niveau `containerd` vers `1.6.25`. CNI reconstruit et `csi-proxy` utilisation `golang1.22.1`.
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.11	1.27.9	1.6.18	1.1.2	WindowsMise à jour autonome KB5034439 exclue sur Windows les AMI principales du serveur 2022. La base de connaissances s'applique uniquement aux Windows installations avec une

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
				WinRE partition séparée, qui ne sont incluses dans aucune de nos Windows AMI optimisées Amazon EKS.
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Correction du problème #2042 containers-roadmap qui empêchait les nœuds d'extraire des images Amazon ECR privées.
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1,26

KubernetesVersion**1.26**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.11	1.26.12	1.6.18	1.1.2	WindowsMise à jour autonome KB5034439 exclue sur Windows les AMI principales du serveur 2022. La base de connaissances s'applique uniquement aux Windows installations avec une WinRE partition séparée, qui ne sont incluses dans aucune de nos Windows AMI optimisées Amazon EKS.
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.26.9. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.26.4. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

KubernetesVersion**1.25**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.25-2024.03.12	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.11	1.25.16	1.6.18	1.1.2	WindowsMise à jour autonome KB5034439 exclue sur Windows les AMI principales du serveur 2022. La base de connaissances s'applique uniquement aux Windows installations avec une WinRE partition séparée, qui ne sont incluses dans aucune de nos Windows AMI optimisées Amazon EKS.
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.25.14. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.25.9. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1,24

KubernetesVersion**1.24**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.11	1.24.17	1.6.18	1.1.2	WindowsMise à jour autonome KB5034439 exclue sur Windows les AMI principales du serveur 2022. La base de connaissances s'applique uniquement aux Windows installations avec une WinRE partition séparée, qui ne sont incluses dans aucune de nos Windows AMI optimisées Amazon EKS.
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.24.17. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.24.13. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Version de Kubernetes rétrogradée vers 1.24.7 parce que 1.24.10 a signalé un problème dans kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

AMI Windows Server 2022 Full optimisée pour Amazon EKS

Le tableau ci-dessous répertorie les versions actuelles et précédentes de l'AMI Windows Server 2022 Full optimisée pour Amazon EKS.

Kubernetes version 1,30

KubernetesVersion**1.30**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

KubernetesVersion**1.29**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Mise à niveau containerd vers 1.7.11. Mise à niveau kubelet vers 1.29.3.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Correction d'un bug à cause duquel l'image de pause était supprimée de manière incorrecte par le processus de collecte kubelet des déchets.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1,28

KubernetesVersion**1.28**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1,27

KubernetesVersion**1.27**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Correction du problème #2042 containers-roadmap qui empêchait les nœuds d'extraire des images Amazon ECR privées.
1.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1,26

KubernetesVersion**1.26**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.26.9. Ajout de nouvelles variables d'environnement pour le script d'amorçag

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
				e (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.26.4. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

KubernetesVersion**1.25**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstru

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
				it et `csi-proxy` utilisation `golang1.22.1`.
`1.25-2024.03.12`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2024.02.13`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2024.01.09`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2023.12.12`	`1.25.15`	`1.6.18`	`1.1.2`	
`1.25-2023.11.14`	`1.25.15`	`1.6.18`	`1.1.2`	Comprend des correctifs pour `CVE-2023-5528` .
`1.25-2023.10.19`	`1.25.14`	`1.6.18`	`1.1.2`	Mise à niveau `containerd` vers `1.6.18`. Mise à niveau `kubelet` vers `1.25.14`. Ajout de nouvelles variables d'environnement pour le script d'amorçage (`SERVICE_IPV4_CIDR` et `EXCLUDED_SNAT_CIDRS`).
`1.25-2023.09.12`	`1.25.12`	`1.6.6`	`1.1.2`	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.25.9. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1,24

KubernetesVersion**1.24**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.24.17. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.24.13. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Version de Kubernetes rétrogradée vers 1.24.7 parce que 1.24.10 a signalé un problème dans kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

AMI Windows Server 2019 Core optimisée pour Amazon EKS

Le tableau ci-dessous répertorie les versions actuelles et précédentes de l'AMI Windows Server 2019 Core optimisée pour Amazon EKS.

Kubernetes version 1,30

KubernetesVersion**1.30**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

KubernetesVersion**1.29**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Mise à niveau `containerd` vers `1.7.11`. Mise à niveau `kubelet` vers `1.29.3`.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Mise à niveau `containerd` vers `1.6.28`. CNI reconstruit et `csi-proxy` utilisation `golang1.22.1`.
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Correction d'un bug à cause duquel l'image de pause était supprimée de manière incorrecte par le processus de collecte kubelet des déchets.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1,28

KubernetesVersion**1.28**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1,27

KubernetesVersion**1.27**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Correction du problème #2042 containers-roadmap qui empêchait les nœuds d'extraire des images Amazon ECR privées.
11.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1,26

KubernetesVersion**1.26**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.26.9. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.26.4. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

KubernetesVersion**1.25**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.25.14. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.25.9. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1,24

KubernetesVersion**1.24**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.24.17. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.24.13. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Version de Kubernetes rétrogradée vers 1.24.7 parce que 1.24.10 a signalé un problème dans kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.11.08	1.24.7	1.6.6	1.1.1	

AMI Windows Server 2019 Full optimisée pour Amazon EKS

Le tableau ci-dessous répertorie les versions actuelles et précédentes de l'AMI Windows Server 2019 Full optimisée pour Amazon EKS.

Kubernetes version 1,30

KubernetesVersion**1.30**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

KubernetesVersion**1.29**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Mise à niveau containerd vers 1.7.11. Mise à niveau kubelet vers 1.29.3.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Correction d'un bug à cause duquel l'image de pause était supprimée de manière incorrecte par le processus de collecte kubelet des déchets.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1,28

KubernetesVersion**1.28**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1,27

KubernetesVersion**1.27**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Correction d'une recommandation de sécurité dans kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Correction du problème #2042 containers-roadmap qui empêchait les nœuds d'extraire des images Amazon ECR privées.
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1,26

KubernetesVersion**1.26**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28. Mise à niveau kubelet vers 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.26.9. Ajout de nouvelles variables d'environnement pour le script d'amorçag

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
				e (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.26.4. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

KubernetesVersion**1.25**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstru

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
				it et `csi-proxy` utilisation `golang1.22.1`.
`1.25-2024.03.13`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2024.02.13`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2024.01.09`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2023.12.12`	`1.25.15`	`1.6.18`	`1.1.2`	
`1.25-2023.11.14`	`1.25.15`	`1.6.18`	`1.1.2`	Comprend des correctifs pour `CVE-2023-5528` .
`1.25-2023.10.19`	`1.25.14`	`1.6.18`	`1.1.2`	Mise à niveau `containerd` vers `1.6.18`. Mise à niveau `kubelet` vers `1.25.14`. Ajout de nouvelles variables d'environnement pour le script d'amorçage (`SERVICE_IPV4_CIDR` et `EXCLUDED_SNAT_CIDRS`).
`1.25-2023.09.12`	`1.25.12`	`1.6.6`	`1.1.2`	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.25.9. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour kubelet et kube-proxy en cas de panne de service.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1,24

KubernetesVersion**1.24**

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	Mise à niveau containerd vers 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Mise à niveau containerd vers 1.6.25. CNI reconstruit et csi-proxy utilisation golang1.22.1.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Comprend des correctifs pour CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Mise à niveau containerd vers 1.6.18. Mise à niveau kubelet vers 1.24.17. Ajout de nouvelles variables d'environnement pour le script d'amorçage (SERVICE_IPV4_CIDR et EXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Mise à niveau du plug-in CNI Amazon VPC pour utiliser le connecteur binaire Kubernetes, qui obtient l'adresse IP du Pod à partir du serveur d'API Kubernetes. Demande d'extraction n° 100 fusionnée.

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Comprend des correctifs pour CVE-2023-3676 , CVE-2023-3893 et CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.21	1.24.13	1.6.6	1.1.1	Résolution d'un problème qui entraînait le remplissage incorrect de la liste de recherche des suffixes DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Mise à niveau Kubernetes vers 1.24.13. Ajout du support pour le mappage des ports hôte dans CNI. Demande d'extraction fusionnée #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Correction d'un bogue provoquant le problème de connectivité réseau #1126 sur les pods après le redémarrage du nœud. Ajout d'un nouveau paramètre de configuration du script d'amorçage (ExcludedSnatCIDRs).

Version d'AMI	Version de **kubelet**	Version de **containerd**	Version de **csi-proxy**	Notes de mise à jour
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Ajout d'un mécanisme de récupération pour `kubelet` et `kube-proxy` en cas de panne de service.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Installation d'un plugin gMSA sans domaine pour faciliter l'authentification gMSA pour les conteneurs Windows sur Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Version de Kubernetes rétrogradée vers `1.24.7` parce que `1.24.10` a signalé un problème dans `kube-proxy` .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.12	1.24.7	1.6.6	1.1.1	

Récupération des ID d'AMI Windows optimisées pour Amazon EKS

Vous pouvez récupérer par programmation l'identifiant Amazon Machine Image (AMI) pour les AMI optimisées Amazon EKS en interrogeant l'API AWS Systems Manager Parameter Store. Ce paramètre vous évite de devoir rechercher manuellement les ID d'AMI optimisées pour Amazon EKS. Pour plus d'informations sur l'API Systems Manager Parameter Store, consultez GetParameter. Le principal IAM que vous utilisez doit disposer de l'autorisation IAM `ssm:GetParameter` pour récupérer les métadonnées de l'AMI optimisée pour Amazon EKS.

Vous pouvez récupérer l'ID d'image de la dernière AMI Windows optimisée pour Amazon EKS recommandée avec la commande suivante en utilisant le sous-paramètre `image_id`. Vous pouvez remplacer *`1.30`* par n'importe quelle version Amazon EKS prise en charge et remplacer *`region-code`* par une région prise en charge par Amazon EKS pour laquelle vous souhaitez obtenir l'ID d'AMI. Remplacez *`Core`* par `Full` pour voir l'ID d'AMI Windows Server complet. Pour Kubernetes version `1.24` ou ultérieure, vous pouvez remplacer *`2019`* par `2022` pour voir l'ID d'AMI Windows Server 2022.

```
aws ssm get-parameter --name /aws/service/ami-windows-latest/Windows_Server-2019-
English-Core-EKS_Optimized-1.30/image_id --region region-code --query "Parameter.Value"
 --output text
```

L'exemple qui suit illustre un résultat.

```
ami-1234567890abcdef0
```

Création d'AMI Windows optimisées pour Amazon EKS personnalisées

Vous pouvez utiliser EC2 Image Builder pour créer des AMI Windows personnalisées optimisées pour Amazon EKS avec l'une des options suivantes :

- Utilisation d'une AMI Windows optimisée pour Amazon EKS comme base
- Utilisation du composant de génération géré par Amazon

Vous devez créer votre propre recette Image Builder avec les deux méthodes. Pour plus d'informations, veuillez consulter la rubrique Créer une nouvelle version d'une recette d'image dans le Guide de l'utilisateur Image Builder.

> **Important**
>
> Les composants gérés par Amazon suivants pour `eks` incluent des correctifs pour `CVE-2023-5528`.
>
> - `1.24.3` et ultérieures
> - `1.25.2` et ultérieures
> - `1.26.2` et ultérieures
> - `1.27.0` et ultérieures
> - `1.28.0` et ultérieures

Utilisation d'une AMI Windows optimisée pour Amazon EKS comme base

Cette option est la méthode recommandée pour créer vos AMI Windows personnalisées. Les AMI Windows optimisées pour Amazon EKS que nous fournissons sont mises à jour plus fréquemment que le composant de génération géré par Amazon.

1. Lancez une nouvelle recette Image Builder.
   a. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder.
   b. Dans le panneau de navigation de gauche, choisissez Recettes d'image.
   c. Choisissez Créer une recette d'image.
2. Dans la section Détails de la recette, saisissez un Nom et une Version.
3. Spécifiez l'ID de l'AMI Windows optimisée pour Amazon EKS dans la section Image de base.
   a. Choisissez Saisir un ID d'AMI personnalisé.
   b. Récupérez l'ID d'AMI correspondant à la version du système d'exploitation Windows dont vous avez besoin. Pour de plus amples informations, veuillez consulter Récupération des ID d'AMI Windows optimisées pour Amazon EKS.
   c. Saisissez l'ID d'AMI personnalisé. Si l'ID de l'AMI est introuvable, assurez-vous que l'ID Région AWS de l'AMI correspond à celui Région AWS affiché dans le coin supérieur droit de votre console.
4. (Facultatif) Pour obtenir les dernières mises à jour de sécurité, ajoutez le composant `update-windows` dans la section Composants de génération.

a. Dans la liste déroulante située à droite du champ de recherche Rechercher des composants par nom, sélectionnez Géré par Amazon.

b. Dans le champ de recherche Rechercher des composants par nom, saisissez **`update-windows`**.

c. Cochez la case du résultat de recherche **`update-windows`**. Ce composant inclut les derniers correctifs Windows du système d'exploitation.

5. Complétez les entrées de recette d'image restantes avec les configurations requises. Pour plus d'informations, veuillez consulter la rubrique Créer une nouvelle version d'une recette d'image (console) dans le Guide de l'utilisateur Image Builder.
6. Choisissez Créer une recette.
7. Utilisez la nouvelle recette d'image dans un pipeline d'images nouveau ou existant. Une fois que votre pipeline d'images s'exécute correctement, votre AMI personnalisée sera répertoriée en tant qu'image de sortie et sera prête à être utilisée. Pour plus d'informations, veuillez consulter la rubrique Créer un pipeline d'images à l'aide de l'assistant de la console EC2 Image Builder.

Utilisation du composant de génération géré par Amazon

Lorsque l'utilisation d'une AMI Windows optimisée pour Amazon EKS comme base n'est pas viable, vous pouvez utiliser le composant de génération géré par Amazon à la place. Cette option peut être en retard par rapport aux versions Kubernetes prises en charge les plus récentes.

1. Lancez une nouvelle recette Image Builder.

   a. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder.

   b. Dans le panneau de navigation de gauche, choisissez Recettes d'image.

   c. Choisissez Créer une recette d'image.
2. Dans la section Détails de la recette, saisissez un Nom et une Version.
3. Déterminez l'option que vous utiliserez pour créer votre AMI personnalisée dans la section Image de base :

   - Sélectionnez les images gérées : choisissez Windows pour votre Système d'exploitation (OS) de l'image. Choisissez ensuite l'une des options suivantes pour Origine de l'image.

- Démarrage rapide (géré par Amazon) : dans la liste déroulante Nom de l'image, sélectionnez une version Windows Server prise en charge par Amazon EKS. Pour de plus amples informations, veuillez consulter AMI Windows optimisées pour Amazon EKS.
- Mes images : pour Nom de l'image, sélectionnez l'ARN de votre propre image avec votre propre licence. Les composants Amazon EKS ne peuvent pas déjà être installés sur l'image que vous fournissez.

- Saisir l'ID d'AMI personnalisée : pour l'ID D'AMI, saisissez l'ID de votre AMI avec votre propre licence. Les composants Amazon EKS ne peuvent pas déjà être installés sur l'image que vous fournissez.

4. Dans la section Composants de génération – Windows, procédez comme suit :

   a. Dans la liste déroulante située à droite du champ de recherche Rechercher des composants par nom, sélectionnez Géré par Amazon.

   b. Dans le champ de recherche Rechercher des composants par nom, saisissez **`eks`**.

   c. Cochez la case du résultat de recherche **`eks-optimized-ami-windows`**, même si le résultat renvoyé peut ne pas être la version que vous voulez.

   d. Dans le champ de recherche Rechercher des composants par nom, saisissez **`update-windows`**.

   e. Cochez la case du résultat de recherche update-windows. Ce composant inclut les derniers correctifs Windows du système d'exploitation.

5. Dans la section Composants sélectionnés, procédez comme suit :

   a. Choisissez Options de gestion des versions pour **`eks-optimized-ami-windows`**.

   b. Choisissez Spécifier la version du composant.

   c. Dans le champ Version du composant, saisissez ***`version.x`*** , en remplaçant *`version`* par une version Kubernetes prise en charge. La saisie d'un *`x`* dans le numéro de version indique l'utilisation de la dernière version du composant qui correspond également à la partie de la version que vous définissez explicitement. Faites attention à la sortie de la console, car elle vous indiquera si la version souhaitée est disponible en tant que composant géré. N'oubliez pas que les versions Kubernetes les plus récentes peuvent ne pas être disponibles pour le composant de génération. Pour plus d'informations sur les versions disponibles, consultez Récupération d'informations sur les versions des composants `eks-optimized-ami-windows`.

> **Note**
>
> Les versions suivantes de composants de génération `eks-optimized-ami-windows` nécessitent une version de `eksctl` `0.129` ou inférieure :
>
> - `1.24.0`

6. Complétez les entrées de recette d'image restantes avec les configurations requises. Pour plus d'informations, veuillez consulter la rubrique Créer une nouvelle version d'une recette d'image (console) dans le Guide de l'utilisateur Image Builder.
7. Choisissez Créer une recette.
8. Utilisez la nouvelle recette d'image dans un pipeline d'images nouveau ou existant. Une fois que votre pipeline d'images s'exécute correctement, votre AMI personnalisée sera répertoriée en tant qu'image de sortie et sera prête à être utilisée. Pour plus d'informations, veuillez consulter la rubrique Créer un pipeline d'images à l'aide de l'assistant de la console EC2 Image Builder.

Récupération d'informations sur les versions des composants **`eks-optimized-ami-windows`**

Vous pouvez récupérer des informations spécifiques concernant ce qui est installé avec chaque composant. Par exemple, vous pouvez vérifier quelle version de `kubelet` est installée. Les composants subissent des tests fonctionnels sur les versions des systèmes d'exploitation Windows prises en charge par Amazon EKS. Pour de plus amples informations, veuillez consulter Calendrier des versions. Toutes les autres versions des systèmes d'exploitation Windows qui sont indiquées comme n'étant pas ou plus prises en charge peuvent ne pas être compatibles avec le composant.

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder.
2. Dans le panneau de navigation de gauche, sélectionnez Composants.
3. Dans la liste déroulante située à droite du champ de recherche Rechercher des composants par nom, modifiez M'appartenant en Démarrage rapide (géré par Amazon).
4. Dans la case Rechercher des composants par nom, saisissez **`eks`**.
5. (Facultatif) Si vous utilisez une version récente, triez la colonne Version par ordre décroissant en la choisissant deux fois.
6. Choisissez le lien **`eks-optimized-ami-windows`** avec la version souhaitée.

La Description de la page qui s'affiche contient les informations spécifiques.

Stockage

Ce chapitre couvre les options de stockage pour les clusters Amazon EKS.

Rubriques

Pilote CSI Amazon EBS

Le pilote Container Storage Interface (CSI) Amazon Elastic Block Store (Amazon EBS) gère le cycle de vie des volumes Amazon EBS en tant que stockage pour les volumes Kubernetes que vous créez. Le pilote Amazon EBS CSI crée des volumes Amazon EBS pour les types de volumes suivants : Kubernetes volumes éphémères génériques et volumes persistants.

Voici quelques points à prendre en compte lors de l'utilisation du pilote Amazon EBS CSI.

- Le plug-in Amazon EBS CSI nécessite des autorisations IAM pour passer des appels aux AWS API en votre nom. Pour plus d'informations, consultez Création du rôle IAM du pilote CSI Amazon EBS.
- Il n'est pas possible de monter des volumes Amazon EBS sur les Pods Fargate.
- Vous pouvez exécuter le contrôleur CSI Amazon EBS sur les nœuds Fargate, mais le nœud CSI DaemonSet Amazon EBS ne peut fonctionner que sur des instances Amazon EC2.

Le pilote Amazon EBS CSI n'est pas installé lorsque vous créez un cluster pour la première fois. Pour utiliser le pilote, vous devez l'ajouter en tant que module complémentaire d'Amazon EKS ou en tant que module complémentaire autogéré.

- Pour obtenir des instructions sur la façon de l'ajouter en tant que module complémentaire Amazon EKS, consultez Gestion du pilote Amazon EBS CSI en tant que module complémentaire d'Amazon EKS.
- Pour obtenir des instructions sur la façon de l'ajouter en tant qu'installation autogérée, consultez le projet Pilote Amazon EBS Container Storage Interface (CSI) sur GitHub.

Après avoir installé le pilote CSI avec l'une ou l'autre des méthodes, vous pouvez tester la fonctionnalité avec un exemple d'application. Pour plus d'informations, voir Déployez un exemple d'application et vérifiez que le pilote CSI fonctionne.

Création du rôle IAM du pilote CSI Amazon EBS

Le plug-in Amazon EBS CSI nécessite des autorisations IAM pour passer des appels aux AWS API en votre nom. Pour plus d'informations, consultez Configuration de l'autorisation du pilote sur GitHub.

> **Note**
>
> Les Pods auront accès aux autorisations attribuées au rôle IAM, sauf si vous bloquez l'accès à IMDS. Pour plus d'informations, consultez Bonnes pratiques de sécurité pour Amazon EKS.

Prérequis

- Un cluster existant.
- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.

La procédure suivante vous montre comment créer un rôle IAM et y associer la politique gérée par AWS . Pour ce faire, vous pouvez utiliser `eksctl`, la AWS Management Console ou la AWS CLI.

> **Note**
>
> Les étapes spécifiques de cette procédure sont destinées à l'utilisation du pilote en tant que module complémentaire d'Amazon EKS. Différentes étapes sont nécessaires pour utiliser le pilote en tant que module complémentaire autogéré. Pour plus d'informations, consultez Configuration des autorisations de pilote sur GitHub.

eksctl

Pour créer votre rôle IAM de plugin CSI d'Amazon EBS avec **`eksctl`**

1. Créez un rôle IAM et associez une politique. AWS gère une politique AWS gérée ou vous pouvez créer votre propre politique personnalisée. Vous pouvez créer un rôle IAM et associer la politique AWS gérée à l'aide de la commande suivante. Remplacez *`my-cluster`* par le nom de votre cluster. La commande déploie une AWS CloudFormation pile qui crée un rôle IAM et y attache la politique IAM. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

   ```
   eksctl create iamserviceaccount \
       --name ebs-csi-controller-sa \
       --namespace kube-system \
       --cluster my-cluster \
       --role-name AmazonEKS_EBS_CSI_DriverRole \
       --role-only \
       --attach-policy-arn arn:aws:iam::aws:policy/service-role/
   AmazonEBSCSIDriverPolicy \
       --approve
   ```

2. Si vous utilisez une clé KMS personnalisée pour le chiffrement de vos volumes Amazon EBS, personnalisez le rôle IAM en fonction des besoins. Par exemple, procédez comme suit :

   a. Copiez et collez le code suivant dans un nouveau fichier *`kms-key-for-encryption-on-ebs`*`.json`. Remplacez *`custom-key-arn`* par l'ARN de clé KMS personnalisé.

      ```
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "kms:CreateGrant",
              "kms:ListGrants",
              "kms:RevokeGrant"
            ],
            "Resource": ["custom-key-arn"],
            "Condition": {
              "Bool": {
                "kms:GrantIsForAWSResource": "true"
      ```

```
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

b. Créez la politique. Vous pouvez remplacer *KMS_Key_For_Encryption_On_EBS_Policy* par un autre nom. Cependant, si vous le remplacez, assurez-vous de le remplacer également dans les étapes suivantes.

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Attachez la politique IAM au rôle à l'aide de la commande suivante. Remplacez *111122223333* par votre ID de compte. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

AWS Management Console

Pour créer votre rôle IAM dans le plugin Amazon EBS CSI avec le AWS Management Console

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.

3. Sur la page Rôles, choisissez Créer un rôle.
4. Sur la page Select trusted entity (Sélectionnez une entité de confiance), procédez comme suit :
   a. Dans la section Trusted entity type (Type d'entité de confiance), sélectionnez Web identity (Identité web).
   b. Pour Fournisseur d'identité, choisissez l'URL du fournisseur OpenID Connect pour votre cluster (comme indiqué sous Présentation dans Amazon EKS).
   c. Pour Audience, choisissez `sts.amazonaws.com`.
   d. Choisissez Suivant.
5. Sur la page Add permissions (Ajouter des autorisations), procédez comme suit :
   a. Dans la zone Filter policies (Politiques de filtre), saisissez `AmazonEBSCSIDriverPolicy`.
   b. Cochez la case à gauche du `AmazonEBSCSIDriverPolicy` renvoyé dans la recherche.
   c. Choisissez Suivant.
6. Sur la page Name, review, and create (Nommer, vérifier et créer), procédez comme suit :
   a. Pour Role name (Nom de rôle), saisissez un nom unique pour votre rôle, par exemple, ***`AmazonEKS_EBS_CSI_DriverRole`***.
   b. Sous Ajouter des balises (Facultatif), ajoutez des métadonnées au rôle en attachant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.
   c. Sélectionnez Créer un rôle.
7. Une fois le rôle créé, choisissez le rôle dans la console pour l'ouvrir et le modifier.
8. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la politique d'approbation).
9. Trouvez la ligne qui ressemble à la ligne suivante :

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Ajoutez une virgule à la fin de la ligne précédente, puis ajoutez la ligne suivante après la ligne précédente. *region-code*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Remplacez *EXAMPLED539D4633E53DE1B71EXAMPLE* avec l'ID du fournisseur OIDC de votre cluster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:ebs-csi-controller-sa"
```

10. Sélectionnez Update Trust Policy (Mettre à jour la politique d'approbation) pour terminer.
11. Si vous utilisez une clé KMS personnalisée pour le chiffrement de vos volumes Amazon EBS, personnalisez le rôle IAM en fonction des besoins. Par exemple, procédez comme suit :

    a. Dans le panneau de navigation de gauche, choisissez Politiques.

    b. Sur la page Politiques, choisissez Créer une politique.

    c. Sur la page Créer une politique, choisissez l'onglet JSON.

    d. Copiez et collez le code suivant dans l'éditeur, en remplaçant *custom-key-arn* par l'ARN de clé KMS personnalisée.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
```

```
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:DescribeKey"
        ],
        "Resource": ["custom-key-arn"]
    }
  ]
}
```

e. Choisissez Suivant : Balises.

f. Sur la page Ajouter des balises (facultatif), choisissez Suivant : Vérifier.

g. Dans Nom, attribuez un nom unique à votre politique (par exemple ***KMS_Key_For_Encryption_On_EBS_Policy***).

h. Choisissez Créer une politique.

i. Dans le panneau de navigation de gauche, choisissez Rôles.

j. Choisissez le fichier ***AmazoneKS_EBS_CSI_ DriverRole*** dans la console pour l'ouvrir afin de le modifier.

k. Dans la liste déroulante Ajouter des autorisations, choisissez Associer des politiques.

l. Dans la zone Filter policies (Politiques de filtre), saisissez *KMS_Key_For_Encryption_On_EBS_Policy*.

m. Cochez la case à gauche du *KMS_Key_For_Encryption_On_EBS_Policy* renvoyé dans la recherche.

n. Choisissez Attach Policies (Attacher des politiques).

AWS CLI

Pour créer votre rôle IAM dans le plugin Amazon EBS CSI avec le AWS CLI

1. Affichez l'URL du fournisseur OIDC de votre cluster. Remplacez *my-cluster* par le nom de votre cluster. Si la sortie de la commande est None, consultez Prérequis.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

L'exemple qui suit illustre un résultat.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Créez le rôle IAM et attachez-y l'action `AssumeRoleWithWebIdentity`.

   a. Copiez le contenu suivant dans un fichier nommé *aws-ebs-csi-driver-trust-policy*`.json`. Remplacez *111122223333* par votre ID de compte. Remplacez *EXAMPLED539D4633E53DE1B71EXAMPLE* et *region-code* par les valeurs renvoyées par l'étape précédente. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Federated": "arn:aws:iam::111122223333:oidc-provider/
   oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
           "StringEquals": {
             "oidc.eks.region-code.amazonaws.com/
   id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
             "oidc.eks.region-code.amazonaws.com/
   id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
   system:ebs-csi-controller-sa"
           }
         }
       }
     ]
   }
   ```

   b. Créez le rôle. Vous pouvez remplacer *AmazonEKS_EBS_CSI_DriverRole* par un autre nom. Si vous le remplacez, assurez-vous de le remplacer dans les étapes suivantes.

   ```
   aws iam create-role \
     --role-name AmazonEKS_EBS_CSI_DriverRole \
     --assume-role-policy-document file://"aws-ebs-csi-driver-trust-
   policy.json"
   ```

3. Joignez une politique. AWS gère une politique AWS gérée ou vous pouvez créer votre propre politique personnalisée. Attachez la politique AWS gérée au rôle à l'aide de la commande

suivante. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

4. Si vous utilisez une clé KMS personnalisée pour le chiffrement de vos volumes Amazon EBS, personnalisez le rôle IAM en fonction des besoins. Par exemple, procédez comme suit :

   a. Copiez et collez le code suivant dans un nouveau fichier *kms-key-for-encryption-on-ebs*`.json`. Remplacez *custom-key-arn* par l'ARN de clé KMS personnalisé.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
```

```
}
```

b. Créez la politique. Vous pouvez remplacer *KMS_Key_For_Encryption_On_EBS_Policy* par un autre nom. Cependant, si vous le remplacez, assurez-vous de le remplacer également dans les étapes suivantes.

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Attachez la politique IAM au rôle à l'aide de la commande suivante. Remplacez *111122223333* par votre ID de compte. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

Maintenant que vous avez créé le rôle IAM du pilote CSI Amazon EBS, vous pouvez passer à Ajout du module complémentaire du pilote CSI Amazon EBS. Lorsque le plug-in est déployé dans cette procédure, il crée et est configuré pour utiliser un compte de service nommé `ebs-csi-controller-sa`. Le compte de service est lié à un `clusterrole` Kubernetes, qui se voit attribuer les autorisations Kubernetes requises.

Gestion du pilote Amazon EBS CSI en tant que module complémentaire d'Amazon EKS

Pour améliorer la sécurité et réduire le volume de travail, vous pouvez gérer le pilote CSI Amazon EBS en tant que module complémentaire Amazon EKS. Pour plus d'informations sur les modules complémentaires Amazon EKS, consultez Modules complémentaires Amazon EKS. Vous pouvez ajouter le module complémentaire CSI d'Amazon EBS en suivant les étapes de Ajout du module complémentaire du pilote CSI Amazon EBS.

Si vous avez ajouté le module complémentaire CSI d'Amazon EBS, vous pouvez le gérer en suivant les étapes des sections Mise à jour du CSI Amazon EBS en tant que module complémentaire Amazon EKS et Suppression du module complémentaire CSI Amazon EBS.

Prérequis

- Un cluster existant. Pour afficher la version requise de la plateforme, exécutez la commande suivante.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver
```

- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Un rôle IAM du pilote CSI Amazon EBS. Si vous ne remplissez pas cette condition préalable, toute tentative d'installation du module complémentaire et d'exécution de `kubectl describe pvc` fera apparaître `failed to provision volume with StorageClass` ainsi qu'une erreur `could not create volume in EC2: UnauthorizedOperation`. Pour plus d'informations, consultez Création du rôle IAM du pilote CSI Amazon EBS.
- Si vous utilisez une politique `PodSecurityPolicy` restreinte au niveau du cluster, assurez-vous que le module complémentaire dispose d'autorisations suffisantes pour être déployé. Pour connaître les autorisations requises par chaque module complémentairePod, consultez la définition du manifeste du module complémentaire correspondant sur GitHub.

> **Important**
>
> Pour utiliser la fonctionnalité d'instantané du pilote CSI Amazon EBS, vous devez installer la prise d'instantanés externe avant l'installation du module complémentaire. Les composants de la prise d'instantanés externe doivent être installés dans l'ordre suivant :
>
> - `CustomResourceDefinition` (CRD) pour `volumesnapshotclasses`, `volumesnapshots` et `volumesnapshotcontents`
> - RBAC (`ClusterRole`, `ClusterRoleBinding`, etc.)
> - contrôleur de déploiement
>
> Pour plus d'informations, consultez Prise d'instantanés CSI sur GitHub.

Ajout du module complémentaire du pilote CSI Amazon EBS

> **Important**
>
> Avant d'ajouter le pilote Amazon EBS en tant que module complémentaire Amazon EKS, vérifiez qu'aucune version autogérée du pilote n'est déjà installée sur votre cluster. Si c'est le cas, voir Désinstallation d'un pilote CSI Amazon EBS autogéré sur GitHub.

Vous pouvez utiliser `eksctl`, l'AWS Management Console ou l'AWS CLI pour ajouter le module complémentaire CSI Amazon EBS à votre cluster.

eksctl

Pour ajouter le module complémentaire CSI Amazon EBS en utilisant **`eksctl`**

Exécutez la commande suivante. Remplacez *`my-cluster`* par le nom de votre cluster, *`111122223333`* par l'ID de votre compte et *`AmazonEKS_EBS_CSI_DriverRole`* par le nom du rôle IAM créé précédemment. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest)Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
eksctl create addon --name aws-ebs-csi-driver --cluster my-cluster --service-
account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

Si vous supprimez l'option **`--force`** et si des paramètres du module complémentaire Amazon EKS entrent en conflit avec vos paramètres existants, alors la mise à jour du module complémentaire Amazon EKS échoue et vous recevez un message d'erreur pour vous aider à résoudre le conflit. Avant de spécifier cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer, car ces paramètres sont remplacés par cette option. Pour plus d'informations sur les autres options de ce paramètre, consultez Modules complémentaires dans la documentation `eksctl`. Pour plus d'informations sur la gestion des champs Amazon EKS Kubernetes veuillez consulter Gestion des champs Kubernetes.

AWS Management Console

Pour ajouter le module complémentaire CSI Amazon EBS en utilisant l'AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Choisissez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire CSI Amazon EBS.
4. Choisissez l'onglet Modules complémentaires.
5. Choisissez Obtenez plus de modules complémentaires.
6. Sur la page Sélectionner des modules complémentaires, procédez comme suit :
   a. Dans la section Modules complémentaires Amazon EKS, sélectionnez la case à cocher Pilote CSI Amazon EBS.
   b. Choisissez Suivant.
7. Sur la page Configurer les paramètres des modules complémentaires sélectionnés, procédez comme suit :
   a. Sélectionnez la version que vous souhaitez utiliser.
   b. Pour Sélectionner le rôle IAM, sélectionnez le nom d'un rôle IAM auquel vous avez associé la politique IAM du pilote CSI Amazon EBS.
   c. (Facultatif) Vous pouvez développer les paramètres de configuration facultatifs. Si vous sélectionnez Remplacer pour la méthode de résolution des conflits, un ou plusieurs des paramètres du module complémentaire existant peuvent être remplacés par les paramètres du module complémentaire Amazon EKS. Si vous n'activez pas cette option et qu'il y a un conflit avec vos paramètres existants, l'opération échoue. Vous pouvez utiliser le message d'erreur qui en résulte pour résoudre le conflit. Avant de sélectionner cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer vous-même.
   d. Choisissez Suivant.
8. Sur la page Vérifier et ajouter, choisissez Créer. Une fois l'installation du module complémentaire terminée, vous pouvez voir le module complémentaire installé.

AWS CLI

Pour ajouter le module complémentaire CSI Amazon EBS en utilisant l'AWS CLI

Exécutez la commande suivante. Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par l'ID de votre compte, et *AmazonEKS_EBS_CSI_DriverRole* par le nom du rôle créé précédemment. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest)Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole
```

Maintenant que vous avez ajouté le pilote CSI d'Amazon EBS en tant que module complémentaire Amazon EKS, vous pouvez passer à Déployez un exemple d'application et vérifiez que le pilote CSI fonctionne. Cette procédure comprend la configuration de la classe de stockage.

Mise à jour du CSI Amazon EBS en tant que module complémentaire Amazon EKS

Amazon EKS ne met pas automatiquement à jour CSI Amazon EBS pour votre cluster lorsque de nouvelles versions sont publiées ou après la mise à jour de votre cluster vers une nouvelle version mineure de Kubernetes. Pour mettre à jour CSI Amazon EBS sur un cluster existant, vous devez lancer la mise à jour, puis Amazon EKS met à jour le module complémentaire pour vous.

eksctl

Pour mettre à jour le module complémentaire CSI Amazon EBS en utilisant **`eksctl`**

1. Vérifiez la version actuelle de votre module complémentaire CSI Amazon EBS. Remplacez *my-cluster* par le nom de votre cluster.

   ```
   eksctl get addon --name aws-ebs-csi-driver --cluster my-cluster
   ```

   L'exemple qui suit illustre un résultat.

   ```
   NAME                    VERSION                 STATUS  ISSUES  IAMROLE
    UPDATE AVAILABLE
   aws-ebs-csi-driver      v1.11.2-eksbuild.1      ACTIVE  0
     v1.11.4-eksbuild.1
   ```

2. Mettez à jour le module complémentaire vers la version renvoyée sous `UPDATE AVAILABLE` en sortie de l'étape précédente.

```
eksctl update addon --name aws-ebs-csi-driver --version v1.11.4-eksbuild.1 --
cluster my-cluster \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

   Si vous supprimez l'option **`--force`** et si des paramètres du module complémentaire Amazon EKS entrent en conflit avec vos paramètres existants, alors la mise à jour du module complémentaire Amazon EKS échoue et vous recevez un message d'erreur pour vous aider à résoudre le conflit. Avant de spécifier cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer, car ces paramètres sont remplacés par cette option. Pour plus d'informations sur les autres options de ce paramètre, consultez Modules complémentaires dans la documentation `eksctl`. Pour plus d'informations sur la gestion des champs Amazon EKS Kubernetes veuillez consulter Gestion des champs Kubernetes.

AWS Management Console

Pour mettre à jour le module complémentaire CSI Amazon EBS en utilisant l'AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Choisissez le nom du cluster pour lequel vous souhaitez mettre à jour le module complémentaire CSI Amazon EBS.
4. Choisissez l'onglet Modules complémentaires.
5. Choisissez Pilote CSI Amazon EBS.
6. Choisissez Modifier.
7. Sur la page Configuration du pilote CSI Amazon EBS, procédez comme suit :
   a. Sélectionnez la version que vous souhaitez utiliser.
   b. Pour Sélectionner le rôle IAM, sélectionnez le nom d'un rôle IAM auquel vous avez associé la politique IAM du pilote CSI Amazon EBS.

c. (Facultatif) Vous pouvez développer les paramètres de configuration facultatifs et les modifiez selon vos besoins.

d. Sélectionnez Enregistrer les modifications.

AWS CLI

Pour mettre à jour le module complémentaire CSI Amazon EBS en utilisant l'AWS CLI

1. Vérifiez la version actuelle de votre module complémentaire CSI Amazon EBS. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --query "addon.addonVersion" --output text
```

L'exemple qui suit illustre un résultat.

```
v1.11.2-eksbuild.1
```

2. Déterminez les versions du module complémentaire CSI Amazon EBS disponibles pour votre version de cluster.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver --kubernetes-
version 1.23 \
  --query "addons[].addonVersions[].[addonVersion,
 compatibilities[].defaultVersion]" --output text
```

L'exemple qui suit illustre un résultat.

```
v1.11.4-eksbuild.1
True
v1.11.2-eksbuild.1
False
```

La version avec `True` en dessous est la version par défaut déployée lorsque le module complémentaire est créé. La version déployée lorsque le module complémentaire est créé peut ne pas être la dernière version disponible. Dans la sortie précédente, la dernière version est déployée lorsque le module complémentaire est créé.

3. Mettez à jour le module complémentaire à la version avec `True` renvoyé en sortie de l'étape précédente. Vous pouvez également mettre à jour vers une version ultérieure si elle a été renvoyée en sortie.

```
aws eks update-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --addon-version v1.11.4-eksbuild.1 \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --resolve-
conflicts PRESERVE
```

   L'option *PRESERVE* (PRÉSERVER) permet de conserver tous les paramètres personnalisés que vous avez définis pour le module complémentaire. Pour plus d'informations sur d'autres options de ce paramètre, consultez update-addon dans la référence des lignes de commande Amazon EKS. Pour plus d'informations sur la gestion de la configuration des modules complémentaires Amazon EKS, consultez Gestion des champs Kubernetes.

Suppression du module complémentaire CSI Amazon EBS

Vous avez deux options pour supprimer un module complémentaire Amazon EKS.

- Conserver le logiciel complémentaire sur votre cluster : cette option supprime la gestion de tous les paramètres par Amazon EKS. Elle supprime également la possibilité pour Amazon EKS de vous informer des mises à jour et de mettre automatiquement à jour le module complémentaire Amazon EKS après le lancement d'une mise à jour. Toutefois, elle préserve le logiciel complémentaire sur votre cluster. Cette option fait du module complémentaire une installation autogérée plutôt qu'un module complémentaire Amazon EKS. Avec cette option, il n'y a pas de temps d'arrêt pour le module complémentaire. Les commandes dans cette procédure utilisent cette option.
- Supprimer entièrement le logiciel complémentaire de votre cluster : nous vous recommandons de supprimer le module complémentaire Amazon EKS de votre cluster uniquement si aucune ressource de votre cluster n'en dépend. Pour effectuer cette option, supprimez `--preserve` à partir de la commande que vous utilisez dans cette procédure.

Si le module complémentaire est associé à un compte IAM, le compte IAM n'est pas supprimé.

Vous pouvez utiliser `eksctl`, l'AWS Management Console, ou l'AWS CLI pour supprimer le module complémentaire CSI Amazon EBS.

eksctl

Pour supprimer le module complémentaire CSI Amazon EBS en utilisant **`eksctl`**

Remplacez *`my-cluster`* avec le nom de votre cluster, puis exécutez la commande suivante.

```
eksctl delete addon --cluster my-cluster --name aws-ebs-csi-driver --preserve
```

AWS Management Console

Pour supprimer le module complémentaire CSI Amazon EBS en utilisant l'AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Choisissez le nom du cluster pour lequel vous souhaitez supprimer le module complémentaire CSI Amazon EBS.
4. Choisissez l'onglet Modules complémentaires.
5. Choisissez Pilote CSI Amazon EBS.
6. Sélectionnez Remove (Supprimer).
7. Dans la boîte de dialogue Supprimer : aws-ebs-csi-driver confirmation, procédez comme suit :
   a. Si vous voulez qu'Amazon EKS cesse de gérer les paramètres du module complémentaire, sélectionnez Conserver sur le cluster. Faites ceci si vous voulez retenir le logiciel du module complémentaire sur votre cluster. Ceci afin que vous puissiez gérer vous-même tous les paramètres du module complémentaire.
   b. Saisissez **`aws-ebs-csi-driver`**.
   c. Sélectionnez Remove (Supprimer).

AWS CLI

Pour supprimer le module complémentaire CSI Amazon EBS en utilisant l'AWS CLI

Remplacez *`my-cluster`* avec le nom de votre cluster, puis exécutez la commande suivante.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver --
preserve
```

Déployez un exemple d'application et vérifiez que le pilote CSI fonctionne

Vous pouvez tester la fonctionnalité du pilote CSI avec un exemple d'application. Cette rubrique en montre un exemple, mais vous pouvez également effectuer les opérations suivantes :

- Déployez un exemple d'application qui utilise la prise d'instantanés externe pour créer des instantanés de volume. Pour de plus amples informations, veuillez consulter Instantanés de volume sur GitHub.
- Déployez un exemple d'application qui utilise le redimensionnement des volumes. Pour de plus amples informations, veuillez consulter Redimensionnement de volume sur GitHub.

Cette procédure utilise l'exemple d'approvisionnement dynamique de volume du référentiel GitHub du pilote Amazon EBS Container Storage Interface (CSI) pour utiliser un volume Amazon EBS alloué dynamiquement.

1. Clonez le référentiel GitHub du pilote Amazon EBS Container Storage Interface (CSI) sur votre système local.

   ```
   git clone https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git
   ```

2. Accédez à l'exemple de répertoire `dynamic-provisioning`.

   ```
   cd aws-ebs-csi-driver/examples/kubernetes/dynamic-provisioning/
   ```

3. (Facultatif) Le fichier `manifests/storageclass.yaml` provisionne les volumes gp2 Amazon EBS par défaut. Pour utiliser des volumes `gp3` à la place, ajoutez `type: gp3` à `manifests/storageclass.yaml`.

   ```
   echo "parameters:
     type: gp3" >> manifests/storageclass.yaml
   ```

4. Déployez la classe de stockage `ebs-sc`, la revendication de volume persistant `ebs-claim` et l'exemple d'application `app` à partir du répertoire `manifests`.

   ```
   kubectl apply -f manifests/
   ```

5. Décrivez la classe de stockage `ebs-sc`.

```
kubectl describe storageclass ebs-sc
```

L'exemple qui suit illustre un résultat.

```
Name:            ebs-sc
IsDefaultClass:  No
Annotations:     kubectl.kubernetes.io/last-applied-
configuration={"apiVersion":"storage.k8s.io/v1","kind":"StorageClass","metadata":
{"annotations":{},"name":"ebs-
sc"},"provisioner":"ebs.csi.aws.com","volumeBindingMode":"WaitForFirstConsumer"}

Provisioner:           ebs.csi.aws.com
Parameters:            <none>
AllowVolumeExpansion:  <unset>
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     WaitForFirstConsumer
Events:                <none>
```

> **Note**
>
> La classe de stockage utilise le mode de liaison de volume `WaitForFirstConsumer`. Cela signifie que les volumes ne sont pas alloués dynamiquement jusqu'à ce qu'un Pod fasse une demande de volume persistant. Pour plus d'informations, consultez Mode de liaison de volume dans la documentation Kubernetes.

6. Observez les Pods dans l'espace de noms par défaut. Après quelques minutes, le statut du Pod `app` devient `Running`.

```
kubectl get pods --watch
```

Saisissez `Ctrl+C` pour revenir à une invite de shell.

7. Répertoriez les volumes persistants dans l'espace de noms par défaut. Recherchez un volume persistant avec la revendication `default/ebs-claim`.

```
kubectl get pv
```

L'exemple qui suit illustre un résultat.

```
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY
   STATUS   CLAIM               STORAGECLASS   REASON   AGE
pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a   4Gi        RWO            Delete
   Bound    default/ebs-claim   ebs-sc                  30s
```

8. Décrivez le volume persistant. Remplacez pvc-*37717cd6-d0dc-11e9-b17f-06fad4858a5a* par la valeur de la sortie de l'étape précédente.

```
kubectl describe pv pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
```

L'exemple qui suit illustre un résultat.

```
Name:              pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
Labels:            <none>
Annotations:       pv.kubernetes.io/provisioned-by: ebs.csi.aws.com
Finalizers:        [kubernetes.io/pv-protection external-attacher/ebs-csi-aws-com]
StorageClass:      ebs-sc
Status:            Bound
Claim:             default/ebs-claim
Reclaim Policy:    Delete
Access Modes:      RWO
VolumeMode:        Filesystem
Capacity:          4Gi
Node Affinity:
  Required Terms:
    Term 0:        topology.ebs.csi.aws.com/zone in [region-code]
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            ebs.csi.aws.com
    VolumeHandle:      vol-0d651e157c6d93445
    ReadOnly:          false
    VolumeAttributes:      storage.kubernetes.io/
csiProvisionerIdentity=1567792483192-8081-ebs.csi.aws.com
Events:                <none>
```

L'ID de volume Amazon EBS correspond à la valeur de `VolumeHandle` dans la sortie précédente.

9. Vérifiez que le Pod écrit des données sur le volume.

```
kubectl exec -it app -- cat /data/out.txt
```

L'exemple qui suit illustre un résultat.

```
Wed May 5 16:17:03 UTC 2021
Wed May 5 16:17:08 UTC 2021
Wed May 5 16:17:13 UTC 2021
Wed May 5 16:17:18 UTC 2021
[...]
```

10. Une fois que vous avez terminé, supprimez les ressources pour cet exemple d'application.

```
kubectl delete -f manifests/
```

Foire aux questions sur la migration vers Amazon EBS CSI

> **Important**
>
> Si vous avez des Pods en cours d'exécution sur un cluster en version `1.22` ou une version antérieure, vous devez installer le pilote Amazon EBS CSI avant de mettre à jour votre cluster en version `1.23` afin d'éviter toute interruption de service.

La fonctionnalité de migration de l'interface de stockage de conteneurs Amazon EBS (CSI) transfère la responsabilité de la gestion des opérations de stockage du fournisseur de stockage EBS dans l'arborescence d'Amazon EBS vers le pilote CSI d'Amazon EBS.

Que sont les pilotes CSI ?

Pilotes CSI :

- remplacez les pilotes de stockage Kubernetes « dans l'arborescence » qui existent dans code source du projet Kubernetes.
- travailler avec des fournisseurs de stockage, tels qu'Amazon EBS.
- fournissez un modèle de plug-in simplifié qui facilite la tâche des fournisseurs de stockage comme AWS pour la publication des fonctionnalités et le maintien de la prise en charge, sans dépendre du cycle de sortie Kubernetes.

Pour plus d'informations, consultez , Introduction à SQSKubernetes dans la documentation CSI.

Qu'est-ce que la migration vers CSI ?

La fonction de migration vers CSI de Kubernetes déplace la responsabilité de la gestion des opérations de stockage des plug-ins de stockage existants dans l'arborescence, tels que `kubernetes.io/aws-ebs`, aux pilotes CSI correspondants. Les objets existants `StorageClass`,`PersistentVolume` e t`PersistentVolumeClaim` (PVC) continuent de fonctionner tant que le pilote CSI correspondant est installé. Lorsque la fonctionnalité est activée :

- Les charges de travail existantes qui utilisent des PVC continuent de fonctionner comme d'habitude.
- Kubernetes transmet le contrôle de toutes les opérations de gestion du stockage aux pilotes CSI.

Pour plus d'informations, veuillez consulter Kubernetes`1.23` :Kubernetes Mise à jour de l'état de migration dans l'arborescence vers CSI sur le Blog Kubernetes.

Pour vous aider à migrer du plug-in intégré à l'arborescence vers les pilotes CSI, les drapeaux `CSIMigration` et `CSIMigrationAWS` sont activés par défaut sur la version `1.23` d'Amazon EKS et les clusters de versions ultérieures. Ces indicateurs permettent à votre cluster de traduire les API de l'arborescence en leurs API CSI équivalents. Ces drapeaux sont placés sur le plan de contrôle Kubernetes géré par Amazon EKS et dans les paramètres `kubelet` configurés dans les AMI optimisées pour Amazon EKS. Si vous disposez de Pods à l'aide de volumes Amazon EBS dans votre cluster, vous devrez installer le pilote Amazon EBS CSI avant de mettre à jour votre cluster vers la version **`1.23`**. Dans le cas contraire, les opérations de volume telles que le provisionnement et le montage risquent de ne pas fonctionner correctement. Pour plus d'informations, consultez Pilote CSI Amazon EBS.

> **Note**
>
> Le fournisseur de `StorageClass` dans l'arborescence est nommé `kubernetes.io/aws-ebs`. Le fournisseur Amazon EBS CSI `StorageClass` est nommé `ebs.csi.aws.com`.

Puis-je monter les volumes **kubernetes.io/aws-ebs StorageClass** dans la version **1.23** et les versions ultérieures des clusters ?

Oui, tant que le Pilote CSI Amazon EBS est installé. Pour une nouvelle version `1.23` et pour les versions ultérieures de clusters, nous vous recommandons d'installer le pilote Amazon EBS CSI dans le cadre de votre processus de création de cluster. Nous recommandons également d'utiliser uniquement `StorageClasses` basé sur le kit du fournisseur `ebs.csi.aws.com`.

Si vous avez mis à jour votre plan de contrôle de cluster vers la version `1.23`et que vous n'avez pas encore mis à jour vos nœuds `1.23`, alors les drapeaux `CSIMigration`, `CSIMigrationAWS` et `kubelet` ne sont pas activés. Dans ce cas, le pilote intégré à l'arborescence est utilisé pour monter les volumes basés sur `kubernetes.io/aws-ebs`. Le pilote Amazon EBS CSI doit cependant être installé, afin de garantir que Pods utilisant les volumes basés sur `kubernetes.io/aws-ebs` peuvent être planifiés. Le pilote est également requis pour garantir le succès des autres opérations de volume.

Puis-je provisionner les volumes **kubernetes.io/aws-ebs StorageClass** sur Amazon EKS **1.23** et sur les versions ultérieures des clusters ?

Oui, tant que le Pilote CSI Amazon EBS est installé.

Le fournisseur **kubernetes.io/aws-ebs StorageClass** a-t-il déjà été supprimé d'Amazon EKS ?

Le fournisseur `kubernetes.io/aws-ebs` `StorageClass` et le type de volume `awsElasticBlockStore` ne sont plus pris en charge. Toutefois, il n'est pas prévu de les supprimer. Ces ressources sont traitées comme faisant partie de l'API Kubernetes.

Comment puis-je installer le pilote CSI Amazon EBS ?

Nous vous recommandons d'installer le module complémentaire Amazon EKS du pilote CSI Amazon EBS. Lorsqu'une mise à jour est requise pour le module complémentaire Amazon EKS, vous devez lancer la mise à jour, puis Amazon EKS mettra à jour le module complémentaire Amazon EKS à votre place. Si vous souhaitez gérer le pilote vous-même, vous pouvez l'installer en utilisant l'open source les charts de Helm.

> **Important**
>
> Le pilote Kubernetes dans l'arborescence Amazon EBS s'exécute sur le kit plan de contrôle Kubernetes. Il utilise les autorisations IAM attribuées à Rôle IAM de cluster Amazon EKS pour fournir des volumes Amazon EBS. Le pilote CSI Amazon EBS s'exécute sur des nœuds. Le pilote requière les autorisations IAM pour provisionner des volumes. Pour plus d'informations, consultez Création du rôle IAM du pilote CSI Amazon EBS.

Comment puis-je vérifier si le pilote Amazon EBS CSI est installé dans mon cluster ?

Pour déterminer si le pilote est installé sur votre cluster, exécutez la commande suivante :

```
kubectl get csidriver ebs.csi.aws.com
```

Pour vérifier si cette installation est gérée par Amazon EKS, exécutez la commande suivante :

```
aws eks list-addons --cluster-name my-cluster
```

Amazon EKS empêchera-t-il une mise à jour du cluster vers la version **1.23** si je n'ai pas déjà installé le pilote Amazon EBS CSI ?

Non.

Que se passe-t-il si j'oublie d'installer le pilote Amazon EBS CSI avant de mettre à jour mon cluster vers la version 1.23 ? Puis-je installer le pilote après avoir mis à jour mon cluster ?

Oui, mais les opérations de volume nécessitant le pilote Amazon EBS CSI échoueront après la mise à jour de votre cluster jusqu'à ce que le pilote soit installé.

Qu'est-ce que le **StorageClass** par défaut appliqué dans la nouvelle version d'Amazon EKS **1.23** et les versions ultérieures des clusters ?

La valeur par défaut du comportement de `StorageClass` reste inchangée. À chaque nouveau cluster, Amazon EKS applique un `kubernetes.io/aws-ebs` basé sur `StorageClass` nommé`gp2`. Nous n'avons pas l'intention de supprimer ce `StorageClass` à partir de clusters

nouvellement créés. Séparé du cluster `StorageClass` par défaut, si vous créez un `StorageClass` basé sur `ebs.csi.aws.com` sans spécifier de type de volume, le pilote Amazon EBS CSI utilisera `gp3` par défaut.

Amazon EKS apportera-t-il des modifications à **`StorageClasses`** déjà présent dans mon cluster existant lorsque je mets à jour mon cluster vers la version **`1.23`** ?

Non.

Comment puis-je migrer un volume persistant à partir du **`kubernetes.io/aws-ebs`StorageClass** pour **`ebs.csi.aws.com`**à l'aide des instantanés ?

Pour migrer un volume persistant, consultez Migration de clusters Amazon EKS depuis des volumes EBS gp2 vers gp3 sur le blogAWS.

Comment puis-je modifier un volume Amazon EBS à l'aide d'annotations ?

À partir de `aws-ebs-csi-driverv1.19.0-eksbuild.2`, vous pouvez modifier les volumes Amazon EBS à l'aide d'annotations dans leur fichier `PersistentVolumeClaim` s (PVC). La nouvelle fonctionnalité de modification du volume est implémentée sous la forme d'un sidecar supplémentaire, appelé `volumemodifier`. Pour plus d'informations, consultez la section Simplification de la migration et de la modification des volumes Amazon EBS sur Kubernetes à l'aide du pilote EBS CSI sur le blog de AWS.

La migration est-elle prise en charge pour les charges de travail Windows ?

Oui. Si vous installez le pilote Amazon EBS CSI à l'aide des Charts de Helm open source, définissez `node.enableWindows` pour `true`. Ce paramètre est défini par défaut dans le cas d'une installation du pilote Amazon EBS CSI en tant que module complémentaire. Lors de la création de `StorageClasses`, définissez le `fsType` vers un système de fichiers Windows, tel que`ntfs`. Les opérations de volume pour les charges de travail Windows sont ensuite migrées vers le pilote Amazon EBS CSI de la même manière que pour les charges de travail Linux.

Pilote CSI Amazon EFS

Amazon Elastic File System (Amazon EFS) fournit un stockage de fichiers entièrement élastique sans serveur pour vous permettre de partager des données de fichiers sans provisionner ni gérer la capacité et les performances de stockage. Le pilote Amazon EFS Container Storage Interface (CSI)

fournit une interface CSI qui permet aux Kubernetes clusters exécutés de gérer le cycle AWS de vie des systèmes de fichiers Amazon EFS. Cette rubrique vous montre comment déployer le pilote Amazon EFS CSI dans votre cluster Amazon EKS.

Considérations

- Le pilote Amazon EFS CSI n'est pas compatible avec les images de conteneurs basées sur Windows.
- Vous ne pouvez pas utiliser le provisionnement dynamique pour les volumes persistants avec les nœuds Fargate, mais vous pouvez utiliser le provisionnement statique.
- Le provisionnement dynamique nécessite le pilote `1.2` ou une version ultérieure. Vous pouvez utiliser le provisionnement statique pour les volumes persistants en utilisant la version `1.1` du pilote sur n'importe quelle version de cluster Amazon EKS prise en charge.
- La version `1.3.2` ou ultérieure de ce pilote prend en charge l'architecture Arm64, y compris les instances basées sur Amazon EC2 Graviton.
- Les versions `1.4.2` et ultérieures de ce pilote prennent en charge l'utilisation de FIPS pour le montage de systèmes de fichiers.
- Prenez note des quotas de ressources pour Amazon EFS. Par exemple, un quota de 1 000 points d'accès peut être créé pour chaque système de fichiers Amazon EFS. Pour plus d'informations, consultez les quotas de ressources Amazon EFS que vous ne pouvez pas modifier.

Prérequis

- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.

> **Note**
>
> Une Pod exécution permet de monter AWS Fargate automatiquement un système de fichiers Amazon EFS.

Création d'un rôle IAM

Le pilote CSI Amazon EFS nécessite des autorisations IAM pour interagir avec votre système de fichiers. Créez un rôle IAM et associez-y la politique AWS gérée requise. Pour ce faire, vous pouvez utiliser `eksctl`, la AWS Management Console ou la AWS CLI.

> **Note**
>
> Les étapes spécifiques de cette procédure sont destinées à l'utilisation du pilote en tant que module complémentaire d'Amazon EKS. Pour plus de détails sur les installations autogérées, voir Configurer l'autorisation du pilote sur GitHub.

eksctl

Création de votre rôle IAM de plugin CSI d'Amazon EFS à l'aide du code **`eksctl`**

Exécutez la commande suivante pour créer le rôle IAM. Remplacez *`my-cluster`* par le nom de votre cluster et *`AmazonEKS_EFS_CSI_DriverRole`* par le nom de votre rôle.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
```

```
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEFSCSIDriverPolicy \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

AWS Management Console

Pour créer votre rôle IAM de pilote CSI Amazon EFS avec le AWS Management Console

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Sur la page Rôles, choisissez Créer un rôle.
4. Sur la page Select trusted entity (Sélectionnez une entité de confiance), procédez comme suit :
   a. Dans la section Trusted entity type (Type d'entité de confiance), sélectionnez Web identity (Identité web).
   b. Pour Fournisseur d'identité, choisissez l'URL du fournisseur OpenID Connect pour votre cluster (comme indiqué sous Présentation dans Amazon EKS).
   c. Pour Audience, choisissez `sts.amazonaws.com`.
   d. Choisissez Suivant.
5. Sur la page Add permissions (Ajouter des autorisations), procédez comme suit :
   a. Dans la zone Filter policies (Politiques de filtre), saisissez *`AmazonEFSCSIDriverPolicy`*.
   b. Cochez la case à gauche du *`AmazonEFSCSIDriverPolicy`* renvoyé dans la recherche.
   c. Choisissez Suivant.
6. Sur la page Name, review, and create (Nommer, vérifier et créer), procédez comme suit :
   a. Pour Role name (Nom de rôle), saisissez un nom unique pour votre rôle, par exemple, ***`AmazonEKS_EFS_CSI_DriverRole`***.

b. Sous Ajouter des balises (Facultatif), ajoutez des métadonnées au rôle en attachant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.

c. Sélectionnez Créer un rôle.

7. Une fois le rôle créé, choisissez le rôle dans la console pour l'ouvrir et le modifier.
8. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la politique d'approbation).
9. Trouvez la ligne qui ressemble à la ligne suivante :

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Ajoutez la ligne suivante au-dessus de la ligne précédente. *region-code*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Remplacez *EXAMPLED539D4633E53DE1B71EXAMPLE* avec l'ID du fournisseur OIDC de votre cluster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:efs-csi-*",
```

10. Modifiez l'opérateur `Condition` de `"StringEquals"` en `"StringLike"`.
11. Sélectionnez Update Trust Policy (Mettre à jour la politique d'approbation) pour terminer.

AWS CLI

Pour créer votre rôle IAM de pilote CSI Amazon EFS avec le AWS CLI

1. Affichez l'URL du fournisseur OIDC de votre cluster. Remplacez *my-cluster* par le nom de votre cluster. Si la sortie de la commande est `None`, consultez Prérequis.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

L'exemple qui suit illustre un résultat.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Créez le rôle IAM qui accorde l'action `AssumeRoleWithWebIdentity`.

   a. Copiez le contenu suivant dans un fichier nommé *aws-efs-csi-driver-trust-policy*`.json`. Remplacez *111122223333* par votre ID de compte. Remplacez *EXAMPLED539D4633E53DE1B71EXAMPLE* et *region-code* par les valeurs renvoyées par l'étape précédente. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le para`rn:aws:`. `arn:aws-us-gov:`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:efs-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

   b. Créez le rôle. Vous pouvez changer le nom de *AmazonEKS_EFS_CSI_DriverRole*. Dans ce cas, veillez à le changer également dans les étapes suivantes.

```
aws iam create-role \
  --role-name AmazonEKS_EFS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-
policy.json"
```

3. Associez la politique AWS gérée requise au rôle à l'aide de la commande suivante. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name AmazonEKS_EFS_CSI_DriverRole
```

Installation du pilote CSI Amazon EFS

Nous vous recommandons d'installer le pilote CSI Amazon EFS en utilisant le module complémentaire Amazon EKS. Pour ajouter un module complémentaire Amazon EKS à votre cluster, voir Création d'un module complémentaire. Pour plus d'informations sur les modules complémentaires, voir Modules complémentaires Amazon EKS. Si vous ne pouvez pas utiliser le module complémentaire Amazon EKS, nous vous encourageons à signaler un problème expliquant pourquoi vous ne pouvez pas le faire dans le référentiel GitHub de la feuille de route des conteneurs (français non garanti).

Sinon, si vous souhaitez une installation autogérée du pilote CSI Amazon EFS, voir Installation sur GitHub.

Création d'un système de fichiers Amazon EFS

Pour créer un système de fichiers Amazon EFS, voir Création d'un système de fichiers Amazon EFS pour Amazon EKS sur GitHub.

Déploiement d'un exemple d'application

Vous pouvez déployer une variété d'applications types et les modifier selon vos besoins. Pour de plus amples informations, voir Exemples sur GitHub.

Pilote CSI Amazon FSx pour Lustre

Le pilote FSx for Lustre Container Storage Interface (CSI) fournit une interface CSI qui permet aux clusters Amazon EKS de gérer le cycle de vie des systèmes de fichiers FSx for Lustre. Pour plus d'informations, consultez le Guide de l'utilisateur FSx pour Lustre.

Cette rubrique explique comment déployer le pilote CSI FSx for Lustre sur votre cluster Amazon EKS et vérifier qu'il fonctionne. Nous vous recommandons d'utiliser la dernière version du pilote. Pour les versions disponibles, consultez la matrice de compatibilité des spécifications CSI sur GitHub.

> **Note**
>
> Le pilote n'est pas pris en charge sur Fargate.

Pour la description détaillée des paramètres disponibles et des exemples complets démontrant les fonctions du pilote, consultez le projet de pilote FSx for Lustre Container Storage Interface (CSI) sur GitHub.

Prérequis

Vous devez disposer de ce qui suit :

- Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.

Les procédures suivantes vous aident à créer un cluster de test simple avec le pilote CSI FSx pour Lustre afin que vous puissiez voir comment il fonctionne. Nous déconseillons l'utilisation du cluster

de test pour les charges de travail de production. Pour ce tutoriel, nous vous recommandons d'utiliser les *example values*, sauf lorsqu'il est noté de les remplacer. Vous pouvez remplacer n'importe quel *example value* lors des étapes propres à un cluster de production. Nous vous recommandons d'effectuer toutes les étapes dans le même terminal, car les variables sont définies et utilisées tout au long des étapes et n'existent pas dans des terminaux différents.

Pour déployer le pilote CSI FSx for Lustre dans un cluster Amazon EKS

1. Définissez quelques variables à utiliser lors des étapes restantes. Remplacez-le *my-csi-fsx-cluster* par le nom du cluster de test que vous souhaitez créer et *region-code* par le Région AWS nom dans lequel vous souhaitez créer votre cluster de test.

   ```
   export cluster_name=my-csi-fsx-cluster
   export region_code=region-code
   ```

2. Créez un cluster de test.

   ```
   eksctl create cluster \
     --name $cluster_name \
     --region $region_code \
     --with-oidc \
     --ssh-access \
     --ssh-public-key my-key
   ```

   L'approvisionnement de cluster dure plusieurs minutes. Lors de la création du cluster, vous verrez plusieurs lignes de sortie. La dernière ligne de sortie est similaire à celle de l'exemple suivant.

   ```
   [#]  EKS cluster "my-csi-fsx-cluster" in "region-code" region is ready
   ```

3. Créez un compte Kubernetes de service pour le pilote et associez la politique `AmazonFSxFullAccess` AWS gérée au compte de service à l'aide de la commande suivante. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le para`rn:aws:`. `arn:aws-us-gov:`

   ```
   eksctl create iamserviceaccount \
     --name fsx-csi-controller-sa \
     --namespace kube-system \
     --cluster $cluster_name \
     --attach-policy-arn arn:aws:iam::aws:policy/AmazonFSxFullAccess \
   ```

```
  --approve \
  --role-name AmazonEKSFSxLustreCSIDriverFullAccess \
  --region $region_code
```

Vous verrez plusieurs lignes de sortie lorsque le compte de service sera créé. Les dernières lignes de sortie sont similaires à celle de l'exemple suivant.

```
[#]  1 task: {
    2 sequential sub-tasks: {
        create IAM role for serviceaccount "kube-system/fsx-csi-controller-sa",
        create serviceaccount "kube-system/fsx-csi-controller-sa",
    } }
[#]  building iamserviceaccount stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  deploying stack "eksctl-my-csi-fsx-cluster-addon-iamserviceaccount-kube-
system-fsx-csi-controller-sa"
[#]  waiting for CloudFormation stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  created serviceaccount "kube-system/fsx-csi-controller-sa"
```

Notez le nom de la AWS CloudFormation pile qui a été déployée. Dans l'exemple de sortie ci-dessus, la pile est nommée `eksctl-my-csi-fsx-cluster-addon-iamserviceaccount-kube-system-fsx-csi-controller-sa`.

4. Installez le pilote avec la commande suivante : Remplacez *release-X.XX* par la branche de votre choix. La branche master n'est pas prise en charge car elle peut contenir des fonctionnalités à venir incompatibles avec la version stable du pilote actuellement disponible. Nous vous recommandons d'utiliser la dernière version publiée. Pour obtenir la liste des branches actives, voir aws-fsx-csi-driverci-dessous GitHub.

> **Note**
>
> Vous pouvez afficher le contenu appliqué dans aws-fsx-csi-driver sur GitHub.

```
kubectl apply -k "github.com/kubernetes-sigs/aws-fsx-csi-driver/deploy/kubernetes/
overlays/stable/?ref=release-X.XX"
```

L'exemple qui suit illustre un résultat.

```
serviceaccount/fsx-csi-controller-sa created
serviceaccount/fsx-csi-node-sa created
clusterrole.rbac.authorization.k8s.io/fsx-csi-external-provisioner-role created
clusterrole.rbac.authorization.k8s.io/fsx-external-resizer-role created
clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-external-provisioner-binding
 created
clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-resizer-binding created
deployment.apps/fsx-csi-controller created
daemonset.apps/fsx-csi-node created
csidriver.storage.k8s.io/fsx.csi.aws.com created
```

5. Notez l'ARN du rôle créé. Si vous ne l'avez pas noté plus tôt et qu'il n'est plus disponible dans la AWS CLI sortie, vous pouvez procéder comme suit pour le voir dans le AWS Management Console.

   a. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

   b. Assurez-vous que la console est configurée sur Région AWS celle dans laquelle vous avez créé votre rôle IAM, puis sélectionnez Stacks.

   c. Sélectionnez la pile nommée `eksctl-`*`my-csi-fsx-cluster`*`-addon-iamserviceaccount-kube-system-`*`fsx-csi-controller-sa`*.

   d. Sélectionnez l'onglet Outputs (Sorties). L'ARN Role1 est répertorié sur la page Sorties (1).

6. Appliquez un correctif sur le déploiement du pilote pour ajouter le compte de service que vous avez créé précédemment avec la commande suivante. Remplacez l'ARN par l'ARN que vous avez noté. Remplacez *`111122223333`* par votre ID de compte. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

   ```
   kubectl annotate serviceaccount -n kube-system fsx-csi-controller-sa \
     eks.amazonaws.com/role-
   arn=arn:aws:iam::111122223333:role/AmazonEKSFSxLustreCSIDriverFullAccess --
   overwrite=true
   ```

   L'exemple qui suit illustre un résultat.

   ```
   serviceaccount/fsx-csi-controller-sa annotated
   ```

Pour déployer une classe de stockage Kubernetes, une revendication de volume persistant et une application type afin de vérifier que le pilote CSI fonctionne

Cette procédure utilise le référentiel GitHub du pilote FSx for Lustre Container Storage Interface (CSI) pour utiliser un volume FSx for Lustre alloué dynamiquement.

1. Notez le groupe de sécurité de votre cluster. Vous pouvez le voir dans la section Réseau AWS Management Console sous la section Réseau ou en utilisant la AWS CLI commande suivante.

```
aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

2. Créez un groupe de sécurité pour votre système de fichiers Amazon FSx en fonction des critères indiqués dans Groupes de sécurité Amazon VPC dans le Guide de l'utilisateur Amazon FSx pour Lustre. Pour le VPC, sélectionnez le VPC de votre cluster, comme indiqué dans la section Mise en réseau. Pour « les groupes de sécurité associés à vos clients Lustre », utilisez le groupe de sécurité de votre cluster. Vous pouvez juste laisser les règles sortantes pour autoriser Tout le trafic.
3. Téléchargez le manifeste de la classe de stockage à l'aide de la commande suivante.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/storageclass.yaml
```

4. Modifiez la section des paramètres du fichier `storageclass.yaml`. Remplacez chaque *`example value`* par vos propres valeurs.

```
parameters:
  subnetId: subnet-0eabfaa81fb22bcaf
  securityGroupIds: sg-068000ccf82dfba88
  deploymentType: PERSISTENT_1
  automaticBackupRetentionDays: "1"
  dailyAutomaticBackupStartTime: "00:00"
  copyTagsToBackups: "true"
  perUnitStorageThroughput: "200"
  dataCompressionType: "NONE"
  weeklyMaintenanceStartTime: "7:09:00"
  fileSystemTypeVersion: "2.12"
```

   - **`subnetId`** – L'ID du sous-réseau dans lequel le système de fichiers Amazon FSx pour Lustre doit être créé. Amazon FSx for Lustre n'est pas pris en charge dans toutes les

zones de disponibilité. Ouvrez la console Amazon FSx for Lustre à l'adresse https://console.aws.amazon.com/fsx/ pour confirmer que le sous-réseau que vous souhaitez utiliser se trouve dans une zone de disponibilité prise en charge. Le sous-réseau peut inclure vos nœuds, ou peut être un sous-réseau ou un VPC différent.

- Vous pouvez vérifier la présence des sous-réseaux de nœuds dans le en AWS Management Console sélectionnant le groupe de nœuds dans la section Calculer.
- Si le sous-réseau que vous spécifiez n'est pas le même que celui dans lequel vous avez des nœuds, vos VPC doivent être connectés et vous devez vous assurer que les ports nécessaires sont ouverts dans vos groupes de sécurité.

- **`securityGroupIds`** – L'ID du groupe de sécurité que vous avez créé pour le système de fichiers.
- **`deploymentType`** (facultatif) – Le type de déploiement du système de fichiers. Les valeurs valides sont `SCRATCH_1`, `SCRATCH_2`, `PERSISTENT_1` et `PERSISTENT_2`. Pour plus d'informations sur les types de déploiement, voir Créer votre système de fichiers Amazon FSx for Lustre.
- autres paramètres (facultatif) — Pour plus d'informations sur les autres paramètres, voir Modifier StorageClass surGitHub.

5. Créez le manifeste de la classe de stockage.

```
kubectl apply -f storageclass.yaml
```

L'exemple qui suit illustre un résultat.

```
storageclass.storage.k8s.io/fsx-sc created
```

6. Téléchargez le manifeste de revendication de volume persistant.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/claim.yaml
```

7. (Facultatif) Modifiez le fichier `claim.yaml`. Remplacez *`1200Gi`* par l'une des valeurs d'incrément ci-dessous, en fonction de vos besoins de stockage et du `deploymentType` que vous avez sélectionné à l'étape précédente.

```
storage: 1200Gi
```

- SCRATCH_2 et PERSISTENT – **`1.2 TiB`**, **`2.4 TiB`** ou des incréments de 2,4 Tio au-dessus de 2,4 Tio.
- SCRATCH_1 – **`1.2 TiB`**, **`2.4 TiB`**, **`3.6 TiB`** ou des incréments de 3,6 Tio au-dessus de 3,6 Tio.

8. Créez la revendication de volume persistant.

```
kubectl apply -f claim.yaml
```

L'exemple qui suit illustre un résultat.

```
persistentvolumeclaim/fsx-claim created
```

9. Vérifiez que le système de fichiers est approvisionné.

```
kubectl describe pvc
```

L'exemple qui suit illustre un résultat.

```
Name:          fsx-claim
Namespace:     default
StorageClass:  fsx-sc
Status:        Bound
[...]
```

> **Note**
>
> Le `Status` peut indiquer `Pending` pendant 5-10 minutes avant de devenir `Bound`. Ne passez pas à l'étape suivante tant que le `Status` n'est pas `Bound`. Si le `Status` affiche `Pending` pendant plus de 10 minutes, utilisez des messages d'avertissement dans les `Events` comme référence afin de résoudre tout problème.

10. Déployez un exemple d'application

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/examples/kubernetes/dynamic_provisioning/specs/pod.yaml
```

11. Vérifiez que l'exemple d'application est en cours d'exécution.

```
kubectl get pods
```

L'exemple qui suit illustre un résultat.

```
NAME      READY   STATUS              RESTARTS   AGE
fsx-app   1/1     Running             0          8s
```

12. Vérifiez que le système de fichiers est correctement monté par l'application.

```
kubectl exec -ti fsx-app -- df -h
```

L'exemple qui suit illustre un résultat.

```
Filesystem                  Size  Used Avail Use% Mounted on
overlay                      80G  4.0G   77G   5% /
tmpfs                        64M     0   64M   0% /dev
tmpfs                       3.8G     0  3.8G   0% /sys/fs/cgroup
192.0.2.0@tcp:/abcdef01     1.1T  7.8M  1.1T   1% /data
/dev/nvme0n1p1               80G  4.0G   77G   5% /etc/hosts
shm                          64M     0   64M   0% /dev/shm
tmpfs                       6.9G   12K  6.9G   1% /run/secrets/kubernetes.io/
serviceaccount
tmpfs                       3.8G     0  3.8G   0% /proc/acpi
tmpfs                       3.8G     0  3.8G   0% /sys/firmware
```

13. Vérifiez que les données ont été écrites sur le système de fichiers FSx for Lustre par l'exemple d'application.

```
kubectl exec -it fsx-app -- ls /data
```

L'exemple qui suit illustre un résultat.

```
out.txt
```

Cet exemple de sortie signifie que l'exemple d'application a réussi à écrire le fichier `out.txt` dans le système de fichiers.

> **Note**
>
> Avant de supprimer le cluster, veillez à supprimer le système de fichiers FSx for Lustre. Pour de plus d'informations, veuillez consulter Nettoyage des ressources dans le Guide de l'utilisateur FSx for Lustre.

Pilote Amazon FSx pour NetApp ONTAP CSI

NetApp'sAstra Tridentfournit une orchestration dynamique du stockage à l'aide d'un pilote conforme à la norme CSI (Container Storage Interface). Cela permet aux clusters Amazon EKS de gérer le cycle de vie des volumes persistants (PV) soutenus par Amazon FSx NetApp pour les systèmes de fichiers ONTAP. Pour commencer, consultez la section Utiliser Astra Trident avec Amazon FSx pour NetApp ONTAP dans la documentation. Astra Trident

Amazon FSx for NetApp ONTAP est un service de stockage qui vous permet de lancer et d'exécuter des systèmes de ONTAP fichiers entièrement gérés dans le cloud. ONTAPest une technologie de système de NetApp's fichiers qui fournit un ensemble largement adopté de fonctionnalités d'accès et de gestion des données. FSx for ONTAP fournit les fonctionnalités, les performances et les API des systèmes de NetApp fichiers sur site avec l'agilité, l'évolutivité et la simplicité d'un service entièrement géré. AWS Pour plus d'informations, consultez le Guide de l'utilisateur FSx for ONTAP.

Pilote CSI Amazon FSx pour OpenZFS

Amazon FSx pour OpenZFS est un service de stockage de fichiers entièrement géré qui facilite le transfert de données vers AWS à partir de serveurs de fichiers ZFS sur site ou d'autres serveurs de fichiers basés sur Linux. Vous pouvez le faire sans modifier le code de votre application ni la façon dont vous gérez les données. Il offre un stockage de fichiers hautement fiable, évolutif, efficace et riche en fonctionnalités basé sur le système de fichiers open source OpenZFS. Il associe ces fonctionnalités à l'agilité, à l'évolutivité et à la simplicité d'un service AWS entièrement géré. Pour plus d'informations, consultez le Guide de l'utilisateur Amazon FSx pour OpenZFS.

Le pilote Amazon FSx pour OpenZFS Container Storage Interface (CSI) fournit une interface CSI qui permet aux clusters Amazon EKS de gérer le cycle de vie des volumes Amazon FSx pour OpenZFS. Pour déployer le pilote Amazon FSx pour OpenZFS CSI sur votre cluster Amazon EKS, consultez `aws-fsx-openzfs-csi-driver` sur GitHub.

Pilote CSI Amazon File Cache

Amazon File Cache est un cache haut débit entièrement géré sur AWS qui est utilisé pour traiter les données des fichiers, quel que soit leur emplacement de stockage. Amazon File Cache charge automatiquement les données dans le cache lorsque vous y accédez pour la première fois et les publie lorsqu'elles ne sont pas utilisées. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur Amazon File Cache.

Le pilote Amazon File Cache Container Storage Interface (CSI) fournit une interface CSI qui permet aux clusters Amazon EKS de gérer le cycle de vie des systèmes de fichiers Amazon. Pour déployer le pilote CSI Amazon File Cache sur votre cluster Amazon EKS, consultez `aws-file-cache-csi-driver` sur GitHub.

Mountpoint pour le pilote CSI Amazon S3

Avec le pilote CSI (Container Storage Interface) Mountpoint pour Amazon S3, vos Kubernetes applications peuvent accéder aux objets Amazon S3 via une interface de système de fichiers, ce qui permet d'obtenir un débit agrégé élevé sans modifier le code de l'application. Basé sur Mountpoint pour Amazon S3, le pilote CSI présente un compartiment Amazon S3 comme un volume auquel peuvent accéder les conteneurs dans Amazon EKS et les clusters Kubernetes autogérés. Cette rubrique vous montre comment déployer Mountpoint pour le pilote Amazon S3 CSI dans votre cluster Amazon EKS.

Considérations

- Le Mountpoint pour le pilote CSI Amazon S3 n'est pas actuellement compatible avec les images de conteneurs basées sur Windows.
- Le Mountpoint pour le pilote CSI Amazon S3 ne prend pas en charge AWS Fargate. Cependant, les conteneurs qui s'exécutent dans Amazon EC2 (soit avec Amazon EKS, soit avec une installation Kubernetes personnalisée) sont pris en charge.
- Le Mountpoint pour le pilote CSI Amazon S3 ne prend en charge que l'allocation statique. L'allocation dynamique, ou la création de nouveaux compartiments, n'est pas prise en charge.

> **Note**
>
> Le provisionnement statique fait référence à l'utilisation d'un compartiment Amazon S3 existant spécifié comme étant `bucketName` `volumeAttributes` dans

l'`PersistentVolume`objet. Pour plus d'informations, consultez Allocation statique sur GitHub.

- Les volumes montés avec le Mountpoint pour le pilote CSI Amazon S3 ne prennent pas en charge toutes les fonctionnalités du système de fichiers POSIX. Pour plus de détails sur le comportement du système de fichiers, consultez Mountpoint pour le comportement du système de fichiers Amazon S3 sur GitHub.

Prérequis

- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Version 2.12.3 ou ultérieure de l' AWS CLI installation et de la configuration sur votre appareil ou. AWS CloudShell
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.

Création d'une politique IAM

Le Mountpoint pour le pilote CSI Amazon S3 nécessite des autorisations Amazon S3 pour interagir avec votre système de fichiers. Cette section montre comment créer une politique IAM qui accorde les autorisations nécessaires.

L'exemple de politique IAM suivant suit les recommandations d'autorisation IAM pour Mountpoint. Vous pouvez également utiliser la politique AWS gérée AmazonS3FullAccess, mais cette politique gérée accorde plus d'autorisations que ce qui est nécessaireMountpoint.

Pour plus d'informations sur les autorisations recommandées pour Mountpoint, consultez Autorisations IAM d'Mountpoint sur GitHub.

Création d'une politique IAM à l'aide de la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Sur la page Politiques, choisissez Créer une politique.
4. Dans Éditeur de politique, choisissez JSON.
5. Sous Éditeur de politique, copiez et collez ce qui suit :

> **Important**
>
> Remplacez `DOC-EXAMPLE-BUCKET1` par votre propre nom de compartiment Amazon S3.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Sid": "MountpointFullBucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
            ]
        },
        {
            "Sid": "MountpointFullObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ]
        }
   ]
}
```

Les compartiments de répertoire, introduits avec la classe de stockage Amazon S3 Express One Zone, utilisent un mécanisme d'authentification différent de celui des compartiments à usage général. Au lieu d'utiliser `s3:*` des actions, vous devez utiliser l'`s3express:CreateSession`action. Pour plus d'informations sur les compartiments de répertoire, consultez la section Buckets de répertoire dans le guide de l'utilisateur Amazon S3.

Vous trouverez ci-dessous un exemple de politique de moindre privilège que vous utiliseriez pour un bucket de répertoire.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3express:CreateSession",
            "Resource": "arn:aws:s3express:aws-region:111122223333:bucket/DOC-
EXAMPLE-BUCKET1--az_id--x-s3"
        }
    ]
}
```

6. Choisissez Suivant.
7. Sur la page Vérifier et créer, donnez un nom à votre politique. Cet exemple de démonstration utilise le nom `AmazonS3CSIDriverPolicy`.
8. Choisissez Créer une politique.

Création d'un rôle IAM

Le Mountpoint pour le pilote CSI Amazon S3 nécessite des autorisations Amazon S3 pour interagir avec votre système de fichiers. Cette section montre comment créer un rôle IAM pour déléguer ces autorisations. Pour créer ce rôle, vous pouvez utiliser `eksctl`, la console IAM ou AWS CLI.

> **Note**
>
> La politique IAM `AmazonS3CSIDriverPolicy` a été créée dans la section précédente.

eksctl

Pour créer votre rôle IAM de Mountpoint pour le pilote CSI Amazon S3 avec **`eksctl`**

Pour créer le rôle IAM et le compte de service Kubernetes, exécutez les commandes suivantes. Ces commandes attachent également la politique IAM `AmazonS3CSIDriverPolicy` au rôle, annotent le compte de service Kubernetes (`s3-csi-controller-sa`) avec l'Amazon Resource Name (ARN) du rôle IAM et ajoutent le nom du compte de service Kubernetes à la politique d'approbation du rôle IAM.

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

IAM console

Pour créer votre rôle IAM de pilote CSI Mountpoint pour Amazon S3 avec AWS Management Console

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Sur la page Rôles, choisissez Créer un rôle.
4. Sur la page Select trusted entity (Sélectionnez une entité de confiance), procédez comme suit :

    a. Dans la section Trusted entity type (Type d'entité de confiance), sélectionnez Web identity (Identité web).

    b. Pour Fournisseur d'identité, choisissez l'URL du fournisseur OpenID Connect pour votre cluster (comme indiqué sous Présentation dans Amazon EKS).

Si aucune URL ne s'affiche, consultez la section Conditions préalables.

c. Pour Audience, choisissez `sts.amazonaws.com`.

d. Choisissez Suivant.

5. Sur la page Add permissions (Ajouter des autorisations), procédez comme suit :

a. Dans la zone Filter policies (Politiques de filtre), saisissez **`AmazonS3CSIDriverPolicy`**.

> **Note**
>
> Cette politique a été créée dans la section précédente.

b. Cochez la case à gauche du résultat `AmazonS3CSIDriverPolicy` renvoyé par la recherche.

c. Choisissez Suivant.

6. Sur la page Name, review, and create (Nommer, vérifier et créer), procédez comme suit :

a. Pour Role name (Nom de rôle), saisissez un nom unique pour votre rôle, par exemple, **`AmazonEKS_S3_CSI_DriverRole`**.

b. Sous Ajouter des balises (Facultatif), ajoutez des métadonnées au rôle en attachant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.

c. Sélectionnez Créer un rôle.

7. Une fois le rôle créé, choisissez le rôle dans la console pour l'ouvrir et le modifier.

8. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la politique d'approbation).

9. Trouvez la ligne qui ressemble à ce qui suit :

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Ajoutez une virgule à la fin de la ligne précédente, puis ajoutez la ligne suivante. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Remplacez *`EXAMPLED539D4633E53DE1B71EXAMPLE`* avec l'ID du fournisseur OIDC de votre cluster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:s3-csi-*"
```

10. Remplacez l'opérateur `Condition` par `"StringEquals"` au lieu de `"StringLike"`.
11. Sélectionnez Update Trust Policy (Mettre à jour la politique d'approbation) pour terminer.

AWS CLI

Pour créer votre rôle IAM de pilote CSI Mountpoint pour Amazon S3 avec AWS CLI

1. Affichez l'URL du fournisseur OIDC de votre cluster. Remplacez *my-cluster* par le nom de votre cluster. Si la sortie de la commande est `None`, consultez Prérequis.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

L'exemple qui suit illustre un résultat.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Créez le rôle IAM, en accordant au compte de service Kubernetes l'action `AssumeRoleWithWebIdentity`.

   a. Copiez le contenu suivant dans un fichier nommé *aws-s3-csi-driver-trust-policy*`.json`. Remplacez *111122223333* par votre ID de compte. Remplacez *EXAMPLED539D4633E53DE1B71EXAMPLE* et *region-code* par les valeurs renvoyées par l'étape précédente.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
```

```
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:s3-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Créez le rôle. Vous pouvez changer le nom de *AmazonEKS_S3_CSI_DriverRole*. Dans ce cas, veillez à le changer également dans les étapes suivantes.

```
aws iam create-role \
  --role-name AmazonEKS_S3_CSI_DriverRole \
  --assume-role-policy-document file://"aws-s3-csi-driver-trust-policy.json"
```

3. Attachez la politique IAM créée précédemment au rôle à l'aide de la commande suivante.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonS3CSIDriverPolicy \
  --role-name AmazonEKS_S3_CSI_DriverRole
```

> **Note**
>
> La politique IAM `AmazonS3CSIDriverPolicy` a été créée dans la section précédente.

4. Sautez cette étape si vous installez le pilote en tant que module complémentaire d'Amazon EKS. Pour les installations autogérées du pilote, créez des comptes de service Kubernetes annotés avec l'ARN du rôle IAM que vous avez créé.

a. Enregistrez le contenu suivant dans un fichier nommé *mountpoint-s3-service-account*`.yaml`. Remplacez *111122223333* par votre ID de compte.

```
---
apiVersion: v1
kind: ServiceAccount
metadata:
```

```
  labels:
    app.kubernetes.io/name: aws-mountpoint-s3-csi-driver
  name: mountpoint-s3-csi-controller-sa
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn:
arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

b. Créez le compte de service Kubernetes sur votre cluster. Le compte de service Kubernetes (`mountpoint-s3-csi-controller-sa`) est annoté avec le rôle IAM que vous avez créé nommé *AmazonEKS_S3_CSI_DriverRole*.

```
kubectl apply -f mountpoint-s3-service-account.yaml
```

> **Note**
>
> Lorsque vous déployez le plugin dans cette procédure, il crée et est configuré pour utiliser un compte de service nommé `s3-csi-driver-sa`.

Installation de Mountpoint pour le pilote Amazon S3 CSI

Vous pouvez installer le Mountpoint pour le pilote CSI Amazon S3 via le module complémentaire Amazon EKS. Vous pouvez utiliser`eksctl`, le AWS Management Console, ou le AWS CLI pour ajouter le module complémentaire à votre cluster.

Vous pouvez éventuellement installer le Mountpoint pilote CSI Amazon S3 en tant qu'installation autogérée. Pour savoir comment procéder à une installation autogérée, consultez Installation sur GitHub.

eksctl

Pour ajouter le module complémentaire Amazon S3 CSI à l'aide de **`eksctl`**

Exécutez la commande suivante. Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par l'ID de votre compte et *AmazonEKS_S3_CSI_DriverRole* par le nom du rôle IAM créé précédemment.

```
eksctl create addon --name aws-mountpoint-s3-csi-driver --cluster my-cluster --
service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
 --force
```

Si vous supprimez l'option **--force** et si des paramètres du module complémentaire Amazon EKS entrent en conflit avec vos paramètres existants, alors la mise à jour du module complémentaire Amazon EKS échoue et vous recevez un message d'erreur pour vous aider à résoudre le conflit. Avant de spécifier cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer, car ces paramètres sont remplacés par cette option. Pour plus d'informations sur les autres options de ce paramètre, consultez Modules complémentaires dans la documentation `eksctl`. Pour plus d'informations sur la gestion des champs Amazon EKS Kubernetes veuillez consulter Gestion des champs Kubernetes.

AWS Management Console

Pour ajouter le module complémentaire CSI Mountpoint pour Amazon S3 à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Choisissez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire CSI Mountpoint pour Amazon S3.
4. Choisissez l'onglet Modules complémentaires.
5. Choisissez Obtenez plus de modules complémentaires.
6. Sur la page Sélectionner des modules complémentaires, procédez comme suit :
   a. Dans la section Amazon EKS-Addons, cochez la case pour le Mountpointpilote Amazon S3 CSI.
   b. Choisissez Suivant.
7. Sur la page Configurer les paramètres des modules complémentaires sélectionnés, procédez comme suit :
   a. Sélectionnez la version que vous souhaitez utiliser.
   b. Pour Sélectionner un rôle IAM, sélectionnez le nom d'un rôle IAM auquel vous avez associé la Mountpoint politique IAM du pilote CSI Amazon S3.

c. (Facultatif) Vous pouvez développer les paramètres de configuration facultatifs. Si vous sélectionnez Remplacer pour la méthode de résolution des conflits, un ou plusieurs des paramètres du module complémentaire existant peuvent être remplacés par les paramètres du module complémentaire Amazon EKS. Si vous n'activez pas cette option et qu'il y a un conflit avec vos paramètres existants, l'opération échoue. Vous pouvez utiliser le message d'erreur qui en résulte pour résoudre le conflit. Avant de sélectionner cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer vous-même.

d. Choisissez Suivant.

8. Sur la page Vérifier et ajouter, choisissez Créer. Une fois l'installation du module complémentaire terminée, vous pouvez voir le module complémentaire installé.

AWS CLI

Pour ajouter le module complémentaire CSI Mountpoint pour Amazon S3 à l'aide du AWS CLI

Exécutez la commande suivante. Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par l'ID de votre compte, et *AmazonEKS_S3_CSI_DriverRole* par le nom du rôle créé précédemment.

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

Configuration de Mountpoint pour Amazon S3

Dans la plupart des cas, vous pouvez configurer Mountpoint pour Amazon S3 avec seulement un nom de compartiment. Pour obtenir des instructions sur la configuration de Mountpoint pour Amazon S3, consultez Configuration de Mountpoint pour Amazon S3 sur GitHub.

Déploiement d'un exemple d'application

Vous pouvez déployer l'allocation statique au pilote sur un compartiment Amazon S3 existant. Pour plus d'informations, consultez Allocation statique sur GitHub.

Suppression Mountpoint du pilote CSI pour Amazon S3

Vous avez deux options pour supprimer un module complémentaire Amazon EKS.

- Conserver le logiciel complémentaire sur votre cluster : cette option supprime la gestion de tous les paramètres par Amazon EKS. Elle supprime également la possibilité pour Amazon EKS de vous informer des mises à jour et de mettre automatiquement à jour le module complémentaire Amazon EKS après le lancement d'une mise à jour. Toutefois, elle préserve le logiciel complémentaire sur votre cluster. Cette option fait du module complémentaire une installation autogérée plutôt qu'un module complémentaire Amazon EKS. Avec cette option, il n'y a pas de temps d'arrêt pour le module complémentaire. Les commandes dans cette procédure utilisent cette option.
- Supprimer entièrement le logiciel complémentaire de votre cluster : nous vous recommandons de supprimer le module complémentaire Amazon EKS de votre cluster uniquement si aucune ressource de votre cluster n'en dépend. Pour effectuer cette option, supprimez `--preserve` à partir de la commande que vous utilisez dans cette procédure.

Si le module complémentaire est associé à un compte IAM, le compte IAM n'est pas supprimé.

Vous pouvez utiliser `eksctl` AWS Management Console, le ou AWS CLI pour supprimer le module complémentaire Amazon S3 CSI.

eksctl

Pour supprimer le module complémentaire Amazon S3 CSI à l'aide de **`eksctl`**

Remplacez *`my-cluster`* avec le nom de votre cluster, puis exécutez la commande suivante.

```
eksctl delete addon --cluster my-cluster --name aws-mountpoint-s3-csi-driver --
preserve
```

AWS Management Console

Pour supprimer le module complémentaire Amazon S3 CSI à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.

3. Choisissez le nom du cluster pour lequel vous souhaitez supprimer le module complémentaire CSI Amazon EBS.
4. Choisissez l'onglet Modules complémentaires.
5. Choisissez Mountpointle pilote Amazon S3 CSI.
6. Sélectionnez Remove (Supprimer).
7. Dans la boîte de dialogue de confirmation Remove : aws-mountpoint-s 3-csi-driver, procédez comme suit :

    a. Si vous voulez qu'Amazon EKS cesse de gérer les paramètres du module complémentaire, sélectionnez Conserver sur le cluster. Faites ceci si vous voulez retenir le logiciel du module complémentaire sur votre cluster. Ceci afin que vous puissiez gérer vous-même tous les paramètres du module complémentaire.

    b. Saisissez **`aws-mountpoint-s3-csi-driver`**.

    c. Sélectionnez Remove (Retirer).

AWS CLI

Pour supprimer le module complémentaire Amazon S3 CSI à l'aide du AWS CLI

Remplacez *`my-cluster`* avec le nom de votre cluster, puis exécutez la commande suivante.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-driver --preserve
```

Contrôleur d'instantané CSI

Le contrôleur d'instantané de l'interface de stockage de conteneurs (CSI) permet d'utiliser la fonctionnalité d'instantané dans les pilotes CSI compatibles, tels que le pilote CSI Amazon EBS.

Voici quelques éléments à prendre en compte lors de l'utilisation du contrôleur d'instantané CSI.

- Le contrôleur d'instantané doit être installé aux côtés d'un pilote CSI doté d'une fonctionnalité d'instantané. Le pilote CSI Amazon EBS prend en charge la création d'instantanés Amazon EBS de volumes gérés Amazon EBS CSI. Pour obtenir des instructions d'installation, consultez Pilote CSI Amazon EBS.

- Kubernetes ne prend pas en charge les instantanés de volumes servis via la migration CSI, tels que les volumes Amazon EBS utilisant un `StorageClass` avec le provisionneur `kubernetes.io/aws-ebs`. Les volumes doivent être créés avec un `StorageClass` qui fait référence au provisionneur de pilote CSI, `ebs.csi.aws.com`. Pour en savoir plus sur la migration CSI, consultez Foire aux questions sur la migration vers Amazon EBS CSI.

Nous vous recommandons d'installer le contrôleur d'instantané CSI via le module complémentaire géré Amazon EKS. Pour ajouter un module complémentaire Amazon EKS à votre cluster, voir Création d'un module complémentaire. Pour plus d'informations sur les modules complémentaires, voir Modules complémentaires Amazon EKS.

Sinon, si vous voulez une installation autogérée du contrôleur d'instantané CSI Amazon EBS, consultez Utilisation dans Kubernetes `external-snapshotter` en amont sur GitHub.

Mise en réseau d'Amazon EKS

Votre cluster Amazon EKS est créé dans un VPC. La mise en réseau du pod est fournie par le plug-in Container Network Interface (CNI) d'Amazon VPC. Ce chapitre inclut les rubriques suivantes pour en savoir plus sur la mise en réseau de votre cluster.

Rubriques

Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux

Lorsque vous créez un cluster, vous spécifiez un VPC et au moins deux sous-réseaux situés dans des zones de disponibilité différentes. Cette rubrique fournit une vue d'ensemble des exigences et considérations spécifiques à Amazon EKS qui sont requises pour le VPC et les sous-réseaux que vous utilisez avec votre cluster. Si vous n'avez pas de VPC à utiliser avec Amazon EKS, vous pouvez en créer un à l'aide d'un modèle fourni par AWS CloudFormation Amazon EKS. Si vous créez un cluster local ou étendu sur AWS Outposts, consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux plutôt cette rubrique.

Exigences et considérations requises pour le VPC

Lorsque vous créez un cluster, le VPC que vous spécifiez doit répondre aux exigences et aux considérations suivantes :

- Le VPC doit disposer d'un nombre suffisant d'adresses IP disponibles pour le cluster, pour tous les nœuds et pour les autres ressources Kubernetes que vous souhaitez créer. Si le VPC que vous souhaitez utiliser ne dispose pas d'un nombre suffisant d'adresses IP, essayez d'augmenter le nombre d'adresses IP disponibles.

Pour ce faire, vous pouvez mettre à jour la configuration du cluster afin de modifier les sous-réseaux et les groupes de sécurité utilisés par le cluster. Vous pouvez effectuer la AWS Management Console mise à jour à partir de la dernière version de AWS CLI AWS CloudFormation, et `v0.164.0-rc.0` ou d'une `eksctl` version ultérieure. Vous pouvez avoir besoin de procéder ainsi pour fournir aux sous-réseaux davantage d'adresses IP disponibles afin de réussir la mise à niveau d'une version de cluster.

> **Important**
>
> Tous les sous-réseaux que vous ajoutez doivent se trouver dans le même ensemble d'AZ que celui fourni à l'origine lors de la création du cluster. Les nouveaux sous-réseaux doivent répondre à toutes les autres exigences, par exemple disposer d'un nombre suffisant d'adresses IP.
> Par exemple, supposons que vous avez créé un cluster et spécifié quatre sous-réseaux. Dans l'ordre où vous les avez spécifiés, le premier sous-réseau se trouve dans la zone de disponibilité `us-west-2a`, les deuxième et troisième sous-réseaux se trouvent dans la zone de disponibilité `us-west-2b` et le quatrième sous-réseau se trouve dans la zone de disponibilité `us-west-2c`. Si vous souhaitez modifier les sous-réseaux, vous devez fournir au moins un sous-réseau dans chacune des trois zones de disponibilité, et les sous-réseaux doivent se trouver dans le même VPC que les sous-réseaux d'origine.

Si vous avez besoin de plus d'adresses IP que les blocs CIDR du VPC, vous pouvez ajouter des blocs CIDR supplémentaires en associant des blocs CIDR (Routage inter-domaines sans classe) supplémentaires à votre VPC. Vous pouvez associer des blocs d'adresse privés (RFC 1918) et publics (non RFC 1918) à votre VPC avant ou après la création de votre cluster. Un cluster peut prendre jusqu'à cinq heures avant qu'un bloc d'adresse CIDR associé à un VPC ne soit reconnu.

Vous pouvez conserver l'utilisation des adresses IP en utilisant une passerelle de transit avec un VPC de services partagés. Pour plus d'informations, consultez VPC isolés avec services partagés et Modèles de conservation des adresses IP routables Amazon EKS VPC dans un réseau hybride.

- Si vous souhaitez que Kubernetes attribue des adresses `IPv6` à des Pods et des services, associez un bloc d'adresse CIDR `IPv6` à votre VPC. Pour plus d'informations, veuillez consulter la section Associer un bloc d'adresse CIDR `IPv6` à votre VPC dans le Guide de l'utilisateur Amazon VPC.

- Le VPC doit prend en charge le nom d'hôte DNS et la résolution DNS. Sinon, les nœuds ne peuvent pas rejoindre votre cluster. Pour plus d'informations, consultez Attributs DNS pour votre VPC dans le guide de l'utilisateur d'Amazon VPC.
- Le VPC peut nécessiter l'utilisation de points de terminaison VPC. AWS PrivateLink Pour plus d'informations, consultez Exigences et considérations requises pour les sous-réseaux.

Si vous avez créé un cluster avec Kubernetes version `1.14` ou une version antérieure, Amazon EKS a ajouté la balise suivante à votre VPC :

Clé	Valeur
`kubernetes.io/cluster/` *`my-cluster`*	`owned`

Cette identification n'a été utilisée que par Amazon EKS. Vous pouvez supprimer l'identification sans affecter vos services. Elle n'est pas utilisée avec des clusters version `1.15` ou ultérieure.

Exigences et considérations requises pour les sous-réseaux

Lorsque vous créez un cluster, Amazon EKS doit créer entre 2 et 4 interfaces réseau Elastic dans les sous-réseaux que vous spécifiez. Ces interfaces réseau permettent la communication entre votre cluster et votre VPC. Ces interfaces réseau activent également des fonctionnalités Kubernetes telles que `kubectl exec` et `kubectl logs`. Chaque interface réseau créée par Amazon EKS possède le texte `Amazon EKS` *`cluster-name`* dans sa description.

Amazon EKS peut créer ses interfaces réseau dans n'importe quel sous-réseau que vous spécifiez lorsque vous créez un cluster. Vous pouvez modifier les sous-réseaux dans lesquels Amazon EKS crée ses interfaces réseau après la création de votre cluster. Lorsque vous mettez à jour la version Kubernetes d'un cluster, Amazon EKS supprime les interfaces réseau d'origine qu'il a créées et crée de nouvelles interfaces réseau. Ces interfaces réseau peuvent être créées dans les mêmes sous-réseaux que les interfaces réseau d'origine ou dans des sous-réseaux différents des interfaces réseau d'origine. Pour contrôler les sous-réseaux dans lesquels les interfaces réseau sont créées, vous pouvez limiter le nombre de sous-réseaux que vous spécifiez à seulement deux lorsque vous créez un cluster ou mettre à jour les sous-réseaux après la création du cluster.

Exigences requises pour les sous-réseaux des clusters

Les sous-réseaux que vous spécifiez lors de la création ou de la mise à jour d'un cluster doivent répondre aux exigences suivantes :

- Les sous-réseaux doivent comporter chacun au moins six adresses IP à utiliser par Amazon EKS. Toutefois, nous vous recommandons au moins 16 adresses IP.
- Les sous-réseaux ne peuvent pas résider dans AWS Outposts une zone AWS locale ou dans une telle zone. AWS Wavelength Toutefois, si vous les avez dans votre VPC, vous pouvez déployer des nœuds autogérés et des ressources Kubernetes sur ces types de sous-réseaux.
- Les sous-réseaux peuvent être publics ou privés. Toutefois, nous vous recommandons de spécifier des sous-réseaux privés, si possible. Un sous-réseau public est un sous-réseau comportant une table de routage comprenant une route vers une passerelle Internet, tandis qu'un sous-réseau privé est un sous-réseau comportant une table de routage qui n'inclut pas de route vers une passerelle Internet.
- Les sous-réseaux ne peuvent pas résider dans les zones de disponibilité suivantes :

Région AWS	Nom de la région	Identifiants de zone de disponibilité non autorisés
`us-east-1`	USA Est (Virginie du Nord)	`use1-az3`
`us-west-1`	USA Ouest (Californie du Nord)	`usw1-az2`
`ca-central-1`	Canada (Centre)	`cac1-az3`

Utilisation de la famille d'adresses IP par composant

Le tableau suivant indique la famille d'adresses IP utilisée par chaque composant d'Amazon EKS. Vous pouvez utiliser un système de traduction d'adresses réseau (NAT) ou un autre système de compatibilité pour vous connecter à ces composants à partir d'adresses IP sources appartenant à des familles dont la "No" valeur correspond à une entrée de table.

Les fonctionnalités peuvent varier en fonction du paramètre IP family (`ipFamily`) du cluster. Ce paramètre modifie le type d'adresses IP utilisé pour le CIDR bloc Kubernetes attribué àServices. Un

cluster avec la valeur de réglage de IPv4 est appelé unIPv4 cluster, et un cluster avec la valeur de réglage IPv6 est appelé unIPv6 cluster.

Composant	**IPv4**adresses uniquement	**IPv6**adresses uniquement	Adresses à double pile
Point de terminaison public d'API EKS	Oui	Non	Non
Point de terminaison VPC de l'API EKS	Oui	Non	Non
Point de terminaison public de l'API EKS Auth	Oui [1]	Oui [1]	Oui [1]
Point de terminaison VPC de l'API EKS Auth	Oui [1]	Oui [1]	Oui [1]
Point de terminaison public du cluster EKS	Oui	Non	Non
Point de terminaison privé du cluster EKS	Oui [2]	Oui [2]	Non
Sous-réseaux du cluster EKS	Oui [2]	Non	Oui [2]
Adresses IP principal es du nœud	Oui [2]	Non	Oui [2]
CIDRPlage de clusters pour les adresses Service IP	Oui [2]	Oui [2]	Non
PodAdresses IP du VPC CNI	Oui [2]	Oui [2]	Non

> **Note**
>
> [1] Le point de terminaison est à double pile avec à la fois `IPv4` des `IPv6` adresses et. Vos applications extérieures AWS, vos nœuds pour le cluster et vos pods à l'intérieur du cluster peuvent atteindre ce point de terminaison par l'un `IPv4` ou l'autre des moyens`IPv6`.
> [2] Vous choisissez entre un `IPv4` cluster et un `IPv6` cluster dans le paramètre IP family (`ipFamily`) du cluster lorsque vous créez un cluster et cela ne peut pas être modifié. Vous devez plutôt choisir un autre paramètre lorsque vous créez un autre cluster et migrez vos charges de travail.

Exigences requises pour les sous-réseaux des nœuds

Vous pouvez déployer des nœuds et des ressources Kubernetes sur les mêmes sous-réseaux que ceux que vous avez spécifiés lors de la création de votre cluster. Toutefois, cela n'est pas nécessaire. En effet, vous pouvez également déployer des nœuds et des ressources Kubernetes sur des sous-réseaux que vous n'avez pas spécifiés lors de la création du cluster. Si vous déployez des nœuds sur différents sous-réseaux, Amazon EKS ne crée pas d'interfaces réseau de cluster dans ces sous-réseaux. Tout sous-réseau sur lequel vous déployez des nœuds et des ressources Kubernetes doit répondre aux exigences suivantes :

- Les sous-réseaux doivent disposer de suffisamment d'adresses IP disponibles pour déployer tous vos nœuds et ressources Kubernetes.
- Si vous souhaitez que Kubernetes attribue des adresses `IPv6` à des Pods et des services, vous devez disposer d'un bloc d'adresse CIDR `IPv6` et d'un bloc d'adresse CIDR `IPv4` associés à votre sous-réseau. Pour plus d'informations, consultez la section Associer un bloc d'adresse CIDR `IPv6` à votre sous-réseau dans le guide de l'utilisateur d'Amazon VPC. Les tables de routage associées aux sous-réseaux doivent inclure des routes vers les adresses `IPv4` et `IPv6`. Pour plus d'informations, veuillez consulter Routes dans le guide de l'utilisateur d'Amazon VPC. Les pods se voient attribuer uniquement une adresse `IPv6`. Toutefois, les interfaces réseau créées par Amazon EKS pour votre cluster et vos nœuds se voient attribuer une adresse `IPv4` et une adresse `IPv6`.
- Si vous avez besoin d'un accès entrant depuis Internet à vos Pods, assurez-vous de disposer d'au moins un sous-réseau public avec suffisamment d'adresses IP disponibles pour déployer des équilibreurs de charge et des entrées. Vous pouvez déployer des équilibreurs de charge sur des sous-réseaux publics. Les équilibreurs de charge peuvent équilibrer la charge vers des Pods situés dans des sous-réseaux privés ou publics. Nous vous recommandons de déployer vos nœuds sur des sous-réseaux privés, si possible.

- Si vous envisagez de déployer des nœuds sur un sous-réseau public, ce sous-réseau doit attribuer automatiquement des adresses publiques IPv4 ou des adresses IPv6. Si vous déployez des nœuds sur un sous-réseau privé doté d'un bloc d'adresse CIDR IPv6 associé, le sous-réseau privé doit également attribuer automatiquement des adresses IPv6. Si vous avez utilisé un AWS CloudFormation modèle Amazon EKS pour déployer votre VPC après le 26 mars 2020, ce paramètre est activé. Si vous avez utilisé les modèles pour déployer votre VPC avant cette date ou si vous utilisez votre propre VPC, vous devez activer ce paramètre manuellement. Pour plus d'informations, consultez Modifier l'attribut d'adressage IPv4 de votre sous-réseau et Modifier l'attribut d'adressage IPv6 de votre sous-réseau dans le guide de l'utilisateur d'Amazon VPC.
- Si le sous-réseau sur lequel vous déployez un nœud est un sous-réseau privé et que sa table de routage n'inclut pas de route vers un appareil de traduction d'adresses réseau (NAT) (IPv4) ou une passerelle de sortie uniquement (IPv6), ajoutez des points de terminaison d'un VPC à l'aide d' AWS PrivateLink à votre VPC. Les points de terminaison VPC sont nécessaires pour tous les éléments avec Services AWS lesquels vos nœuds Pods doivent communiquer. Les exemples incluent Amazon ECR, Elastic Load Balancing CloudWatch AWS Security Token Service, Amazon et Amazon Simple Storage Service (Amazon S3). Le point de terminaison doit inclure le sous-réseau dans lequel se trouvent les nœuds. Tous ne sont pas Services AWS compatibles avec les points de terminaison VPC. Pour plus d'informations, voir Qu'est-ce que c'est AWS PrivateLink ? et AWS des services qui s'intègrent à AWS PrivateLink. Pour obtenir une liste complète des exigences Amazon EKS, consultez Exigences relatives aux clusters privés.
- Si vous souhaitez déployer des équilibreurs de charge sur un sous-réseau, le sous-réseau doit comporter l'identification suivante :
  - Sous-réseaux privés

Clé	Valeur
kubernetes.io/role/internal-elb	1

  - Sous-réseaux publics

Clé	Valeur
kubernetes.io/role/elb	1

Quand un cluster Kubernetes version `1.18` et antérieure était créé, Amazon EKS ajoutait l'identification suivante à tous les sous-réseaux qui avaient été spécifiés.

Clé	Valeur
`kubernetes.io/cluster/` *`my-cluster`*	`shared`

Désormais, lorsque vous créez un cluster Kubernetes, Amazon EKS n'ajoute pas l'identification à vos sous-réseaux. Si l'identification se trouvait sur des sous-réseaux utilisés par un cluster dont la version était antérieure à `1.19`, l'identification n'était pas automatiquement supprimée des sous-réseaux lors de la mise à jour du cluster vers une version plus récente. La version `2.1.1` ou antérieure du AWS Load Balancer Controller nécessite cette identification. Si vous utilisez une version plus récente du Load Balancer Controller, vous pouvez supprimer l'identification sans interrompre vos services.

Si vous avez déployé un VPC à l'aide de l'un `eksctl`des modèles de AWS CloudFormation VPC Amazon EKS, les règles suivantes s'appliquent :

- À partir du 26 mars 2020 : les adresses `IPv4` publiques sont automatiquement attribuées par des sous-réseaux publics aux nouveaux nœuds déployés sur des sous-réseaux publics.
- Avant le 26 mars 2020 : les adresses `IPv4` publiques ne sont pas automatiquement attribuées par les sous-réseaux publics aux nouveaux nœuds déployés sur des sous-réseaux publics.

Cette modification affecte les nouveaux groupes de nœuds déployés sur des sous-réseaux publics de la manière suivante :

- Groupes de nœuds gérés : si le groupe de nœuds est déployé sur un sous-réseau public à partir du 22 avril 2020, l'affectation automatique des adresses IP publiques doit être activée dans le sous-réseau public. Pour plus d'informations, consultez Modification de l'attribut d'adressage `IPv4` public de votre sous-réseau.
- Groupes de nœuds autogérés Linux, Windows ou Arm : si le groupe de nœuds est déployé sur un sous-réseau public à partir du 26 mars 2020, l'attribution automatique des adresses IP publiques doit être activée dans le sous-réseau public. Sinon, les nœuds doivent être lancés avec une adresse IP publique. Pour plus d'informations, consultez Modification de l'attribut d'adressage `IPv4` public de votre sous-réseau ou Attribution d'une adresse `IPv4` publique lors du lancement d'une instance.

Exigences et considérations requises pour les sous-réseaux partagés

Vous pouvez utiliser Partage de VPC pour partager des sous-réseaux avec d'autres comptes AWS au sein d'un même AWS Organizations. Vous pouvez créer des clusters Amazon EKS dans des sous-réseaux partagés, en tenant compte des points suivants :

- Le propriétaire du sous-réseau VPC doit partager un sous-réseau avec un compte participant avant que ce compte puisse y créer un cluster Amazon EKS.
- Vous ne pouvez pas lancer de ressources en utilisant le groupe de sécurité par défaut du VPC, car il appartient au propriétaire. De plus, les participants ne peuvent pas lancer de ressources avec des groupes de sécurité détenus par d'autres participants ou par le propriétaire.
- Dans un sous-réseau partagé, le participant et le propriétaire contrôlent séparément les groupes de sécurité au sein de leur compte respectif. Le propriétaire du sous-réseau peut voir ces groupes de sécurité créés par les participants, mais ne peut pas exécuter d'actions sur ceux-ci. Si le propriétaire du sous-réseau souhaite supprimer ou modifier ces groupes de sécurité, le participant qui a créé le groupe de sécurité doit effectuer l'action.
- Si un cluster est créé par un participant, il faut tenir compte des éléments suivants :
  - Le rôle IAM de cluster et les rôles IAM de nœud doivent être créés dans ce compte. Pour plus d'informations, consultez Rôle IAM de cluster Amazon EKS et Rôle IAM de nœud Amazon EKS.
  - Tous les nœuds doivent être créés par le même participant, y compris les groupes de nœuds gérés.
- Le propriétaire du VPC partagé ne peut pas afficher, mettre à jour ou supprimer un cluster créé par un participant dans le sous-réseau partagé. Cela s'ajoute aux ressources VPC auxquelles chaque compte a un accès différent. Pour plus d'informations, consultez Responsabilités et autorisations pour les propriétaires et les participants dans le Guide de l'utilisateur Amazon VPC.
- Si vous utilisez la fonctionnalité de mise en réseau personnalisée du Amazon VPC CNI plugin for Kubernetes, vous devez utiliser les mappages d'ID de zone de disponibilité répertoriés dans le compte propriétaire pour créer chaque `ENIConfig`. Pour plus d'informations, consultez Mise en réseau personnalisée pour les pods.

Pour plus d'informations sur le partage de sous-réseaux VPC, consultez Partager votre VPC avec d'autres comptes dans le Guide de l'utilisateur Amazon VPC.

Création d'un VPC pour votre cluster Amazon EKS

Vous pouvez utiliser Amazon Virtual Private Cloud (Amazon VPC) pour lancer AWS des ressources dans un réseau virtuel que vous avez défini. Ce réseau virtuel ressemble beaucoup à un réseau traditionnel que vous pourriez exécuter dans votre propre centre de données. Toutefois, il présente l'avantage d'utiliser l'infrastructure évolutive d'Amazon Web Services. Il est recommandé de bien connaître le service Amazon VPC avant de déployer des clusters Amazon EKS de production. Pour de plus amples informations, consultez le Guide de l'utilisateur Amazon VPC.

Un cluster Amazon EKS, des nœuds et des ressources Kubernetes sont déployés sur un VPC. Si vous souhaitez utiliser un VPC existant avec Amazon EKS, ce VPC doit répondre aux exigences décrites dans Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux. Cette rubrique explique comment créer un VPC répondant aux exigences d'Amazon EKS à l'aide d'un modèle fourni par AWS CloudFormation Amazon EKS. Une fois que vous avez déployé un modèle, vous pouvez consulter les ressources créées par le modèle pour savoir exactement quelles ressources il a créées, et la configuration de ces ressources.

Prérequis

Pour créer un VPC pour Amazon EKS, vous devez disposer des autorisations IAM nécessaires pour créer des ressources Amazon VPC. Ces ressources sont des VPC, des sous-réseaux, des groupes de sécurité, des tables de routage et des routes, ainsi que des passerelles Internet et NAT. Pour de plus amples informations, veuillez consulter Créer un VPC avec un exemple de politique de sous-réseau public dans le Guide de l'utilisateur Amazon VPC et la liste complète des Actions, ressources et clés de condition pour Amazon EC2 dans la Référence de l'autorisation de service.

Vous pouvez créer un VPC avec des sous-réseaux à la fois publics et privés, uniquement avec des sous-réseaux publics ou uniquement avec des sous-réseaux privés.

Public and private subnets

> Ce VPC comporte deux sous-réseaux publics et deux privés. Un sous-réseau public est une table de routage associée comportant une route vers une passerelle Internet. Toutefois, la table de routage d'un sous-réseau privé ne comporte pas de route vers une passerelle Internet. Un sous-réseau public et un sous-réseau privé sont déployés dans la même zone de disponibilité. Les autres sous-réseaux public et privé sont déployés dans une seconde zone de disponibilité, dans la même Région AWS. Nous recommandons cette option pour la plupart des déploiements.
>
> Avec cette option, vous pouvez déployer vos nœuds sur des sous-réseaux privés. Cette option permet à Kubernetes de déployer des équilibreurs de charge dans les sous-réseaux publics qui

peuvent équilibrer la charge du trafic vers les Pods exécutés sur les nœuds figurant dans les sous-réseaux privés. Les adresses IPv4 publiques sont automatiquement attribuées aux nœuds déployés sur des sous-réseaux publics, mais les adresses IPv4 publiques ne sont pas attribuées aux nœuds déployés sur des sous-réseaux privés.

Vous pouvez également attribuer des adresses IPv6 aux nœuds des sous-réseaux publics et privés. Les nœuds des sous-réseaux privés peuvent communiquer avec le cluster et d'autres Services AWS. Les Pods peuvent communiquer vers l'Internet, par le biais d'une passerelle NAT à l'aide d'adresses IPv4 ou d'une passerelle Internet sortante uniquement utilisant des adresses IPv6 qui est déployée dans chaque zone de disponibilité. Un groupe de sécurité est déployé et comporte des règles qui refusent tout trafic entrant provenant de sources autres que le cluster ou les nœuds mais autorise tout le trafic sortant. Les sous-réseaux sont balisés de sorte que Kubernetes puisse y déployer des équilibreurs de charge.

Pour créer votre VPC

1. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
2. Dans la barre de navigation, sélectionnez un produit Région AWS compatible avec Amazon EKS.
3. Sélectionnez Créer une pile, Avec de nouvelles ressources (standard).
4. Sous Prerequisite - Prepare template (Prérequis - Préparer le modèle), assurez-vous que Template is ready (Le modèle est prêt) est sélectionné, puis sous Spécifier un modèle, sélectionnez Amazon S3 URL (URL Amazon S3).
5. Vous pouvez créer un VPC qui prend en charge uniquement IPv4, ou un VPC qui prend en charge IPv4 et IPv6. Collez l'une des URL suivantes dans la zone de texte sous URL Amazon S3 et choisissez Suivant :

   - IPv4

     ```
     https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
     ```

   - IPv4 et IPv6

     ```
     https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-subnets.yaml
     ```

6. Dans la page Spécifier les détails de la pile, entrez les paramètres, puis choisissez Suivant.

   - Nom de la pile : choisissez un nom pour votre pile AWS CloudFormation . Par exemple, vous pouvez utiliser le nom du modèle que vous avez utilisé à l'étape précédente. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
   - VpcBlock: Choisissez une plage d'`IPv4`adresses CIDR pour votre VPC. Chaque nœud, Pod et équilibreur de charge que vous déployez se voit attribuer une adresse `IPv4` à partir de ce bloc. Les valeurs `IPv4` par défaut fournissent suffisamment d'adresses IP pour la plupart des implémentations, mais si ce n'est pas le cas, vous pouvez les modifier. Pour plus d'informations, consultez Dimensionnement des VPC et des sous-réseaux dans le Guide de l'utilisateur Amazon VPC. Vous pouvez également ajouter des blocs d'adresse CIDR supplémentaires au VPC une fois qu'il est créé. Si vous créez un VPC `IPv6`, des plages CIDR `IPv6` vous sont automatiquement attribuées à partir de l'espace Global Unicast Address d'Amazon.
   - PublicSubnet01Block : Spécifiez un bloc `IPv4` CIDR pour le sous-réseau public 1. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier. Si vous créez un VPC `IPv6`, ce bloc est spécifié pour vous dans le modèle.
   - PublicSubnet02Block : Spécifiez un bloc `IPv4` CIDR pour le sous-réseau public 2. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier. Si vous créez un VPC `IPv6`, ce bloc est spécifié pour vous dans le modèle.
   - PrivateSubnet01Block : Spécifiez un bloc `IPv4` CIDR pour le sous-réseau privé 1. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier. Si vous créez un VPC `IPv6`, ce bloc est spécifié pour vous dans le modèle.
   - PrivateSubnet02Block : Spécifiez un bloc `IPv4` CIDR pour le sous-réseau privé 2. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier. Si vous créez un VPC `IPv6`, ce bloc est spécifié pour vous dans le modèle.

7. (Facultatif) Sur la page Configure stack options (Configurer les options de pile), étiquetez vos ressources de pile, puis choisissez Next (Suivant).

8. Sur la page Review (Vérification), choisissez Create stack (Créer une pile).
9. Lorsque votre pile est créée, sélectionnez-la dans la console et choisissez Outputs (Sorties).
10. Enregistrez le VpcIdpour le VPC créé. Vous en avez besoin lors de la création de votre cluster et de vos nœuds.
11. Enregistrez SubnetIdsles sous-réseaux créés et indiquez si vous les avez créés en tant que sous-réseaux publics ou privés. Vous en avez besoin d'au moins deux lors de la création de votre cluster et de vos nœuds.
12. Si vous avez créé un VPC `IPv4`, ignorez cette étape. Si vous avez créé un VPC `IPv6`, vous devez activer l'option d'attribution automatique d'adresses `IPv6` pour les sous-réseaux publics qui ont été créés par le modèle. Ce paramètre est déjà activé pour les sous-réseaux privés. Pour activer le paramètre, effectuez les étapes suivantes :

    a. Ouvrez la console Amazon VPC à l'adresse https://console.aws.amazon.com/vpc/.
    b. Dans le volet de navigation de gauche, sélectionnez Subnets (Sous-réseaux).
    c. Sélectionnez l'un de vos sous-réseaux publics (***stack-name/SubnetPublic01 ou stack-name***/SubnetPublic02 contient le mot public) et choisissez Actions, Modifier les paramètres du sous-réseau.
    d. Choisissez la case à cocher Activer l'attribution automatique d'une adresse **IPv6**, puis choisissez Enregistrer.
    e. Effectuez à nouveau les étapes précédentes pour votre autre sous-réseau public.

Only public subnets

Ce VPC dispose de trois sous-réseaux publics qui sont déployés dans des zones de disponibilité différentes dans une Région AWS. Tous les nœuds se voient attribuer automatiquement des adresses `IPv4` publiques et peuvent envoyer et recevoir du trafic Internet via une passerelle Internet. Un groupe de sécurité est déployé qui refuse tout le trafic entrant et autorise tout le trafic sortant. Les sous-réseaux sont balisés de sorte que Kubernetes puisse y déployer des équilibreurs de charge.

Pour créer votre VPC

1. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
2. Dans la barre de navigation, sélectionnez un produit Région AWS compatible avec Amazon EKS.

3. Choisissez Create stack (Créer une pile), Avec de nouvelles ressources (standard).
4. Sous Prepare template (Préparer le modèle), assurez-vous que Prepare template (Modèle est prêt) est sélectionné, puis sous Template source (Source du modèle), sélectionnez Amazon S3 URL (URL Amazon S3).
5. Collez l'URL suivante dans la zone de texte sous URL Amazon S3 et sélectionnez Next (Suivant) :

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-sample.yaml
```

6. Dans la page Spécifier les détails, entrez les paramètres, puis choisissez Suivant.

   - Nom de la pile : choisissez un nom pour votre pile AWS CloudFormation . Par exemple, vous pouvez l'appeler ***amazon-eks-vpc-sample***. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
   - VpcBlock: Choisissez un bloc CIDR pour votre VPC. Chaque nœud, Pod et équilibreur de charge que vous déployez se voit attribuer une adresse `IPv4` à partir de ce bloc. Les valeurs `IPv4` par défaut fournissent suffisamment d'adresses IP pour la plupart des implémentations, mais si ce n'est pas le cas, vous pouvez les modifier. Pour plus d'informations, consultez Dimensionnement des VPC et des sous-réseaux dans le Guide de l'utilisateur Amazon VPC. Vous pouvez également ajouter des blocs d'adresse CIDR supplémentaires au VPC une fois qu'il est créé.
   - Subnet01Block : spécifiez un bloc d'adresse CIDR pour le sous-réseau 1. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier.
   - Subnet02Block : spécifiez un bloc d'adresse CIDR pour le sous-réseau 2. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier.
   - Subnet03Block : spécifiez un bloc d'adresse CIDR pour le sous-réseau 3. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier.
7. (Facultatif) Sur la page Options (Options), étiquetez les ressources de votre pile. Choisissez Next (Suivant).
8. Sur la page Review (Vérification), choisissez Create (Créer).

9. Lorsque votre pile est créée, sélectionnez-la dans la console et choisissez Outputs (Sorties).
10. Enregistrez le VpcIdpour le VPC créé. Vous en avez besoin lors de la création de votre cluster et de vos nœuds.
11. Enregistrez le SubnetIdspour les sous-réseaux créés. Vous en avez besoin d'au moins deux lors de la création de votre cluster et de vos nœuds.
12. (Facultatif) Tout cluster que vous déployez sur ce VPC peut attribuer des adresses `IPv4` privées à vos Pods et services. Si vous souhaitez que des clusters déployés sur ce VPC attribuent des adresses `IPv6` privées à vos Pods et services, mettez à jour votre VPC, votre sous-réseau, vos tables de routage et vos groupes de sécurité. Pour de plus amples informations, veuillez consulter la section Migrer les VPC existants d'`IPv4` vers `IPv6` du Guide de l'utilisateur Amazon VPC. Amazon EKS requiert que l'option `Auto-assign` d'adresses `IPv6` soit activée pour vos sous-réseaux. Elle est désactivée par défaut.

Only private subnets

Ce VPC dispose de trois sous-réseaux privés qui sont déployés dans des zones de disponibilité différentes dans la Région AWS. Les ressources déployées sur les sous-réseaux ne peuvent pas accéder à Internet, et Internet ne peut pas accéder aux ressources dans les sous-réseaux. Le modèle crée des points de terminaison VPC en utilisant plusieurs Services AWS points AWS PrivateLink auxquels les nœuds ont généralement besoin d'accéder. Si vos nœuds ont besoin d'un accès Internet sortant, vous pouvez ajouter une passerelle NAT publique dans la zone de disponibilité de chaque sous-réseau après la création du VPC. Un groupe de sécurité est créé qui refuse tout trafic entrant, à l'exception des ressources déployées dans les sous-réseaux. Un groupe de sécurité autorise également tout le trafic sortant. Les sous-réseaux sont balisés de sorte que Kubernetes puisse y déployer des équilibreurs de charge internes. Si vous créez un VPC avec cette configuration, consultez Exigences relatives aux clusters privés pour des exigences et des considérations supplémentaires.

Pour créer votre VPC

1. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
2. Dans la barre de navigation, sélectionnez un produit Région AWS compatible avec Amazon EKS.
3. Choisissez Create stack (Créer une pile), Avec de nouvelles ressources (standard).

4. Sous Prepare template (Préparer le modèle), assurez-vous que Prepare template (Modèle est prêt) est sélectionné, puis sous Template source (Source du modèle), sélectionnez Amazon S3 URL (URL Amazon S3).
5. Collez l'URL suivante dans la zone de texte sous URL Amazon S3 et sélectionnez Next (Suivant) :

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-fully-private-vpc.yaml
```

6. Sur la page Spécifier les détails, renseignez les paramètres, puis choisissez Suivant.

   - Nom de la pile : choisissez un nom pour votre pile AWS CloudFormation . Par exemple, vous pouvez l'appeler ***amazon-eks-fully-private-vpc***. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
   - VpcBlock: Choisissez un bloc CIDR pour votre VPC. Chaque nœud, Pod et équilibreur de charge que vous déployez se voit attribuer une adresse `IPv4` à partir de ce bloc. Les valeurs `IPv4` par défaut fournissent suffisamment d'adresses IP pour la plupart des implémentations, mais si ce n'est pas le cas, vous pouvez les modifier. Pour plus d'informations, consultez Dimensionnement des VPC et des sous-réseaux dans le Guide de l'utilisateur Amazon VPC. Vous pouvez également ajouter des blocs d'adresse CIDR supplémentaires au VPC une fois qu'il est créé.
   - PrivateSubnet01Block : Spécifiez un bloc CIDR pour le sous-réseau 1. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier.
   - PrivateSubnet02Block : Spécifiez un bloc CIDR pour le sous-réseau 2. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier.
   - PrivateSubnet03Block : Spécifiez un bloc CIDR pour le sous-réseau 3. La valeur par défaut fournit suffisamment d'adresses IP pour la plupart des implémentations. Cependant, si ce n'est pas le cas, vous pouvez la modifier.
7. (Facultatif) Sur la page Options (Options), étiquetez les ressources de votre pile. Choisissez Next (Suivant).
8. Sur la page Review (Vérification), choisissez Create (Créer).

9. Lorsque votre pile est créée, sélectionnez-la dans la console et choisissez Outputs (Sorties).
10. Enregistrez le VpcIdpour le VPC créé. Vous en avez besoin lors de la création de votre cluster et de vos nœuds.
11. Enregistrez le SubnetIdspour les sous-réseaux créés. Vous en avez besoin d'au moins deux lors de la création de votre cluster et de vos nœuds.
12. (Facultatif) Tout cluster que vous déployez sur ce VPC peut attribuer des adresses `IPv4` privées à vos Pods et services. Si vous souhaitez que des clusters déployés sur ce VPC attribuent des adresses `IPv6` privées à vos Pods et services, vous devez mettre à jour votre VPC, votre sous-réseau, vos tables de routage et vos groupes de sécurité. Pour de plus amples informations, veuillez consulter la section Migrer les VPC existants d'`IPv4` vers `IPv6` du Guide de l'utilisateur Amazon VPC. Amazon EKS requiert que l'option `Auto-assign` d'adresses `IPv6` soit activée pour vos sous-réseaux (elle est désactivée par défaut).

Considérations et exigences relatives aux groupes de sécurité Amazon EKS

Cette rubrique décrit les exigences de groupe de sécurité d'un cluster Amazon EKS.

Lorsque vous créez un cluster, Amazon EKS crée un groupe de sécurité nommé `eks-cluster-sg-`*`my-cluster`*`-`*`uniqueID`*. Ce groupe de sécurité dispose des règles par défaut suivantes :

Type de règle	Protocole	Ports	Source	Destination
Entrant	Tous	Tous	Auto-utilisateur	
Sortant	Tous	Tous		0.0.0.0/0(`IPv4`) ou ::/0 (`IPv6`)

> **Important**
>
> Si votre cluster n'a pas besoin de la règle sortante, vous pouvez la supprimer. Si vous la supprimez, vous devez toujours avoir les règles minimales énumérées dans la section Restriction du trafic du cluster. Si vous supprimez la règle entrante, Amazon EKS la recrée à chaque fois que le cluster est mis à jour.

Amazon EKS ajoute les balises suivantes au groupe de sécurité. Si vous supprimez les balises, Amazon EKS les ajoute à nouveau au groupe de sécurité à chaque fois que le cluster est mis à jour.

Clé	Valeur
kubernetes.io/cluster/ *my-cluster*	owned
aws:eks:cluster-name	*my-cluster*
Name	eks-cluster-sg- *my-cluster* -*uniqueid*

Amazon EKS associe automatiquement ce groupe de sécurité aux ressources suivantes qu'il crée également :

- 2 à 4 interfaces réseau élastiques (désignées pour le reste de ce document sous le nom d'interface réseau) créés lors de la création de votre cluster.
- Interfaces réseau des nœuds dans n'importe quel groupe de nœuds géré que vous créez.

Les règles par défaut permettent à tout le trafic de circuler librement entre votre cluster et vos nœuds, et autorise tout le trafic sortant vers n'importe quelle destination. Lorsque vous créez un cluster, spécifiez (éventuellement) vos propres groupes de sécurité. Si c'est le cas, Amazon EKS associe également les groupes de sécurité que vous spécifiez aux interfaces réseau qu'il crée pour votre cluster. Toutefois, il ne les associe à aucun groupe de nœuds que vous créez.

Vous pouvez déterminer l'ID du groupe de sécurité de votre cluster dans la AWS Management Console sous la section Networking (Mise en réseau) du cluster. Pour ce faire, vous pouvez également exécuter la commande AWS CLI suivante :

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Restriction du trafic de cluster

Si vous devez limiter les ports ouverts entre le cluster et les nœuds, vous pouvez supprimer la règle sortante par défaut et ajouter les règles minimales suivantes qui sont requises pour le cluster. Si vous

supprimez la règle entrante par défaut, Amazon EKS la recrée à chaque fois que le cluster est mis à jour.

Type de règle	Protocole	Port	Destination
Sortant	TCP	443	Groupe de sécurité du cluster
Sortant	TCP	10250	Groupe de sécurité du cluster
Sortant (DNS)	TCP et UDP	53	Groupe de sécurité du cluster

Vous devez également ajouter des règles pour le trafic suivant :

- Tout protocole et port que vos nœuds peuvent utiliser pour la communication entre les nœuds.
- Accès Internet sortant afin que les nœuds puissent accéder aux API Amazon EKS pour l'introspection des clusters et l'enregistrement des nœuds au moment du lancement. Si vos nœuds n'ont pas accès à Internet, consultez Exigences relatives aux clusters privés pour des considérations supplémentaires.
- L'accès du nœud pour récupérer les images de conteneurs depuis Amazon ECR ou d'autres registres de conteneurs nécessite l'utilisation des API dont ils ont besoin pour extraire des images, comme DockerHub. Pour plus d'informations, consultez AWS IP Address Ranges dans le manuel Références générales AWS.
- Accès au nœud Amazon S3.
- Des règles distinctes sont requises pour les adresses `IPv4` et `IPv6`.

Si vous envisagez de limiter les règles, nous vous recommandons de tester minutieusement tous vos Pods avant d'appliquer les règles modifiées à un cluster de production.

Si vous avez initialement déployé un cluster avec Kubernetes version `1.14` et une version plateforme `eks.3` ou antérieure, prenez en considération les éléments suivants :

- Vous pouvez également disposer de groupes de sécurité de nœud et de plan de contrôle. Lorsque ces groupes ont été créés, ils incluaient les règles restreintes répertoriées dans le tableau précédent. Ces groupes de sécurité ne sont plus nécessaires et peuvent être supprimés. Toutefois,

vous devez vous assurer que le groupe de sécurité de votre cluster contient les règles que ces groupes contiennent.

- Si vous avez déployé le cluster directement à l'aide de l'API ou si vous avez utilisé un outil tel que la AWS CLI ou AWS CloudFormation pour créer le cluster et que vous n'avez pas spécifié de groupe de sécurité lors de la création du cluster, le groupe de sécurité par défaut du VPC a été appliqué aux interfaces réseau de cluster créées par Amazon EKS.

Modules complémentaires de mise en réseau d'Amazon EKS

Plusieurs modules complémentaires de mise en réseau sont disponibles pour votre cluster Amazon EKS.

Modules complémentaires intégrés

> **Note**
>
> Si vous créez des clusters autrement qu'à l'aide de la console, chaque cluster sera équipé des versions autogérées des modules complémentaires intégrés. Les versions autogérées ne peuvent pas être gérées à partir du AWS Management Console ou AWS Command Line Interface des SDK. Vous gérez la configuration et les mises à niveau des modules complémentaires autogérés.
> Nous recommandons d'ajouter le type Amazon EKS du module complémentaire à votre cluster au lieu d'utiliser le type autogéré du module complémentaire. Lorsque vous créez des clusters dans la console, des modules complémentaires de type Amazon EKS sont installés.

Amazon VPC CNI plugin for Kubernetes

Ce module complémentaire CNI crée des interfaces réseau élastiques et les attache à vos nœuds Amazon EC2. Le module complémentaire attribue également une adresse privée `IPv4` ou `IPv6` de votre VPC à chaque Pod et service. Par défaut, ce module complémentaire est installé sur votre cluster. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

CoreDNS

CoreDNS est un serveur DNS flexible et extensible qui peut servir de DNS de cluster Kubernetes. CoreDNS fournit une résolution de nom pour tous les Pods présents dans le cluster. Par défaut,

ce module complémentaire est installé sur votre cluster. Pour plus d'informations, consultez Utilisation du module complémentaire CoreDNS Amazon EKS.

`kube-proxy`

Ce module complémentaire maintient les règles du réseau sur vos nœuds Amazon EC2 et permet la communication du réseau à vos Pods. Par défaut, ce module complémentaire est installé sur votre cluster. Pour plus d'informations, consultez Utilisation du module complémentaire Kubernetes `kube-proxy`.

Modules complémentaires AWS réseau en option

AWS Load Balancer Controller

Lorsque vous déployez des objets de Kubernetes service de type`loadbalancer`, le contrôleur crée des équilibreurs de charge AWS réseau. Lorsque vous créez des objets Kubernetes d'entrée, le contrôleur crée des équilibreurs de charge AWS d'application. Nous vous recommandons d'utiliser ce contrôleur pour provisionner Network Load Balancers plutôt que d'utiliser l'ancien contrôleur Cloud Provider intégré à Kubernetes. Pour de plus amples informations, veuillez consulter la documentation AWS Load Balancer Controller.

AWS Contrôleur d'API Gateway

Ce contrôleur vous permet de connecter des services sur plusieurs clusters Kubernetes à l'aide de l'API de passerelle Kubernetes. Le contrôleur connecte les services Kubernetes exécutés sur des instances Amazon EC2, des conteneurs et des fonctions sans serveur à l'aide du service Amazon VPC Lattice. Pour plus d'informations, veuillez consulter la documentation Contrôleur d'API Gateway AWS.

Pour plus d'informations sur les modules complémentaires, voir Modules complémentaires Amazon EKS.

Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS

Le module complémentaire Amazon VPC CNI plugin for Kubernetes est déployé sur chaque nœud Amazon EC2 de votre cluster Amazon EKS. Le module complémentaire crée des interfaces réseau élastiques et les attache à vos nœuds Amazon EC2. Le module complémentaire attribue également une adresse privée `IPv4` ou `IPv6` de votre VPC à chaque Pod et service.

Une version du module complémentaire est déployée avec chaque nœud Fargate de votre cluster, mais vous ne devez pas la mettre à jour sur les nœuds Fargate. D'autres plugins CNI compatibles peuvent être utilisés sur les clusters Amazon EKS, mais il s'agit du seul plugin CNI pris en charge par Amazon EKS.

Le tableau suivant répertorie la dernière version disponible du module complémentaire Amazon EKS pour chaque version de Kubernetes.

Version de Kubernetes	1.30	1.29	1.28	1.27	1.26	1.25	1.24	1.23
Type de version VPC CNI d'Amazon EKS	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild.1

> **Important**
>
> Si vous gérez vous-même ce module complémentaire, les versions répertoriées dans le tableau peuvent ne pas être les mêmes que les versions autogérées disponibles. Pour plus d'informations sur la mise à jour du type autogéré de ce module complémentaire, consultez la rubrique Mise à jour du module complémentaire autogéré.

> **Important**
>
> Pour passer à VPC CNI v1.12.0 ou version ultérieure, vous devez d'abord effectuer la mise à niveau vers VPC CNI v1.7.0. Nous vous recommandons de mettre à jour une version mineure à la fois.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.

- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Un rôle IAM auquel est associée la politique IAM AmazonEKS_CNI_Policy (si votre cluster utilise la famille `IPv4`) ou une politique IPv6 (si votre cluster utilise la famille `IPv6`). Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
- Si vous utilisez la version `1.7.0` ou ultérieure du Amazon VPC CNI plugin for Kubernetes et que vous utilisez des politiques de sécurité de Pod personnalisées, consultez Supprimer la stratégie de sécurité de Pod Amazon EKS par défautPolitique de sécurité de pod.
-

  Important

  Amazon VPC CNI plugin for Kubernetesversions `v1.16.0` pour `v1.16.1` supprimer la compatibilité avec Kubernetes les versions `1.23` et antérieures. La version VPC CNI `v1.16.2` rétablit la compatibilité avec Kubernetes les versions antérieures `1.23` et les spécifications CNI. `v0.4.0`

  Amazon VPC CNI plugin for Kubernetesversions `v1.16.0` pour `v1.16.1` implémenter la version `v1.0.0` de spécification CNI. La spécification CNI `v1.0.0` est prise en charge sur les clusters EKS qui exécutent les Kubernetes versions `v1.24` ou ultérieures. Les versions `v1.16.0` VPC CNI `v1.16.1` et CNI `v1.0.0` ne sont pas prises en charge sur les versions antérieures ou antérieures. Kubernetes `v1.23` Pour plus d'informations sur `v1.0.0` la spécification CNI, voir Spécification de l'interface réseau de conteneurs (CNI) sur

Considérations

- Les versions sont spécifiées comme `major-version.minor-version.patch-version-eksbuild.build-number`.
- Vérifiez la compatibilité des versions pour chaque fonctionnalité

  Certaines fonctionnalités de chaque version Amazon VPC CNI plugin for Kubernetes nécessitent certaines Kubernetes versions. Lorsque vous utilisez différentes fonctionnalités Amazon EKS, si une version spécifique du module complémentaire est requise, celle-ci est indiquée dans la documentation des fonctionnalités. Nous vous recommandons d'utiliser la dernière version, sauf si vous avez une raison précise d'utiliser une version antérieure.

Création du module complémentaire Amazon EKS

Créez le type Amazon EKS du module complémentaire.

1. Déterminez la version du module complémentaire actuellement installée sur votre cluster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

L'exemple qui suit illustre un résultat.

```
v1.16.4-eksbuild.2
```

2. Déterminez le type de module complémentaire installé sur votre cluster. Selon l'outil avec lequel vous avez créé votre cluster, le type de module complémentaire Amazon EKS peut ne pas être actuellement installé sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

```
$ aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster, et vous n'avez donc pas besoin de suivre les étapes restantes de cette procédure. Si une erreur est renvoyée, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Suivez les étapes restantes de cette procédure pour l'installer.

3. Enregistrez la configuration du module complémentaire actuellement installé.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Créez le module complémentaire à l'aide de la AWS CLI. Si vous souhaitez utiliser le AWS Management Console ou `eksctl` pour créer le module complémentaire, consultez Création d'un module complémentaire et spécifiez `vpc-cni` le nom du module complémentaire. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée.

   - Remplacez *my-cluster* par le nom de votre cluster.
   - Remplacez *v1.18.2-eksbuild.1* par la dernière version répertoriée dans le tableau des dernières versions pour la version de votre cluster.

- Remplacez *111122223333* par l'ID de votre compte et *AmazonEKSVPCCNIRole* par le nom d'un rôle IAM que vous avez créé. Pour spécifier un rôle, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un pour votre cluster ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.2-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

Si vous avez appliqué à votre module complémentaire actuel des paramètres personnalisés qui entrent en conflit avec les paramètres par défaut du module complémentaire Amazon EKS, la création peut échouer. Si la création échoue, vous recevez un message d'erreur qui peut vous aider à résoudre le problème. Vous pouvez également ajouter **`--resolve-conflicts OVERWRITE`** à la commande précédente. Cela permet au module complémentaire de remplacer les paramètres personnalisés existants. Une fois que vous avez créé le module complémentaire, vous pouvez le mettre à jour avec vos paramètres personnalisés.

5. Assurez-vous que la dernière version du module complémentaire correspondant à la version de Kubernetes de votre cluster a été ajoutée à votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

La création du module complémentaire peut prendre plusieurs secondes.

L'exemple qui suit illustre un résultat.

```
v1.18.2-eksbuild.1
```

6. Si vous avez personnalisé les paramètres du module complémentaire, avant de créer le module complémentaire Amazon EKS, utilisez la configuration que vous avez enregistrée lors d'une étape précédente pour mettre à jour le module complémentaire Amazon EKS avec vos paramètres personnalisés.
7. (Facultatif) Installez le `cni-metrics-helper` sur votre cluster. Il collecte les informations relatives à l'interface Elastic Network et aux adresses IP, les agrège au niveau du cluster et

publie les statistiques sur Amazon. CloudWatch Pour plus d'informations, consultez cni-metrics-helper on. GitHub

Mise à jour du module complémentaire Amazon EKS

Mettez à jour le type de module complémentaire Amazon EKS. Si vous n'avez pas ajouté le type de module complémentaire Amazon EKS à votre cluster, ajoutez-le ou consultez la rubrique Mise à jour du module complémentaire autogéré au lieu de terminer cette procédure.

1. Déterminez la version du module complémentaire actuellement installée sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

L'exemple qui suit illustre un résultat.

```
v1.16.4-eksbuild.2
```

Si la version renvoyée est identique à la version de Kubernetes de votre cluster dans le tableau des dernières versions, cela signifie que la dernière version est déjà installée sur votre cluster, et vous n'avez donc pas besoin de terminer cette procédure. Si vous recevez une erreur au lieu d'un numéro de version dans votre sortie, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Vous devez créer le module complémentaire avant de pouvoir le mettre à jour à l'aide de cette procédure.

2. Enregistrez la configuration du module complémentaire actuellement installé.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

3. Mettez à jour votre module complémentaire à l'aide de la AWS CLI. Si vous souhaitez utiliser AWS Management Console ou mettre `eksctl` à jour le module complémentaire, consultezMise à jour d'un module complémentaire. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée.

   - Remplacez *my-cluster* par le nom de votre cluster.

- Remplacez *v1.18.2-eksbuild.1* par la dernière version répertoriée dans le tableau des dernières versions pour la version de votre cluster.
- Remplacez *111122223333* par l'ID de votre compte et *AmazonEKSVPCCNIRole* par le nom d'un rôle IAM que vous avez créé. Pour spécifier un rôle, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un pour votre cluster ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- L'option **--resolve-conflicts** *PRESERVE* conserve les valeurs de configuration existantes pour le module complémentaire. Si vous avez défini des valeurs personnalisées pour les paramètres des modules complémentaires et que vous n'utilisez pas cette option, Amazon EKS remplace vos valeurs par ses valeurs par défaut. Si vous utilisez cette option, nous vous recommandons de tester les modifications de champ et de valeur sur un cluster hors production avant de mettre à jour le module complémentaire sur votre cluster de production. Si vous remplacez cette valeur par `OVERWRITE`, tous les paramètres sont remplacés par les valeurs par défaut d'Amazon EKS. Si vous avez défini des valeurs personnalisées pour certains paramètres, il est possible qu'elles soient remplacées par les valeurs par défaut d'Amazon EKS. Si vous remplacez cette valeur par `none`, Amazon EKS ne modifie la valeur d'aucun paramètre, mais la mise à jour risque d'échouer. Si la mise à jour échoue, vous recevez un message d'erreur pour vous aider à résoudre le conflit.
- Si vous ne mettez à jour aucun paramètre de configuration, supprimez **--configuration-values '{*"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}*}'** de la commande. Si vous mettez à jour un paramètre de configuration, remplacez *"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}* par le paramètre que vous souhaitez définir. Dans cet exemple, la variable d'environnement `AWS_VPC_K8S_CNI_EXTERNALSNAT` est définie sur `true`. La valeur que vous spécifiez doit être valide pour le schéma de configuration. Si vous ne connaissez pas le schéma de configuration**aws eks describe-addon-configuration --addon-name vpc-cni --addon-version *v1.18.2-eksbuild.1***, lancez-le en remplaçant *v1.18.2-eksbuild.1* par le numéro de version du module complémentaire dont vous souhaitez consulter la configuration. Le schéma est renvoyé dans la sortie. Si vous disposez déjà d'une configuration personnalisée, que vous souhaitez la supprimer et rétablir les valeurs par défaut d'Amazon EKS pour tous les paramètres, supprimez *"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}* de la commande, de sorte que le champ **{}** soit vide. Pour une explication de chaque paramètre, voir Variables de configuration CNI activées GitHub.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.2-eksbuild.1 \
```

```
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
 \
    --resolve-conflicts PRESERVE --configuration-values '{"env":
{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}}'
```

La mise à jour peut prendre plusieurs secondes.

4. Assurez-vous que la version du module complémentaire a été mise à jour. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

La mise à jour peut prendre plusieurs secondes.

L'exemple qui suit illustre un résultat.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.18.2-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/vpc-
cni/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "serviceAccountRoleArn":
 "arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole",
        "tags": {},
        "configurationValues": "{\"env\":{\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":\"true
\"}}"
    }
}
```

Mise à jour du module complémentaire autogéré

> **Important**
>
> Nous recommandons d'ajouter le type Amazon EKS du module complémentaire à votre cluster au lieu d'utiliser le type autogéré du module complémentaire. Si la différence entre les types ne vous est pas familière, consultez the section called "Modules complémentaires Amazon EKS". Pour plus d'informations sur l'ajout d'un module complémentaire Amazon EKS à votre cluster, consultez the section called "Création d'un module complémentaire". Si vous ne parvenez pas à utiliser le module complémentaire Amazon EKS, nous vous encourageons à signaler les raisons pour lesquelles vous ne pouvez pas utiliser le module complémentaire Amazon EKS dans le GitHub référentiel de feuilles de route pour les conteneurs.

1. Assurez-vous que le type de module complémentaire Amazon EKS est installé sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    addon.addonVersion --output text
   ```

   Si un message d'erreur est renvoyé, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Pour gérer vous-même le module complémentaire, suivez les étapes restantes de cette procédure afin de le mettre à jour. Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster. Pour le mettre à jour, suivez la procédure décrite dans la rubrique Mise à jour d'un module complémentaire plutôt que cette procédure. Si les différences entre les types de modules complémentaires ne vous sont pas familières, consultez Modules complémentaires Amazon EKS.
2. Découvrez quelle version de l'image de conteneur est actuellement installée sur votre cluster.

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
    | cut -d : -f 3
   ```

   L'exemple qui suit illustre un résultat.

   ```
   v1.16.4-eksbuild.2
   ```

Il est possible que votre sortie n'inclue pas le numéro de build.

3. Sauvegardez vos paramètres actuels afin de pouvoir configurer les mêmes paramètres une fois que vous aurez mis à jour votre version.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Pour voir les versions disponibles et vous familiariser avec les modifications apportées à la version vers laquelle vous souhaitez effectuer la mise à jour, consultez la page `releases` sur GitHub. Notez que nous vous recommandons de procéder à la même mise à jour`major`. `minor`. `patch`version répertoriée dans le tableau des dernières versions disponibles, même si des versions ultérieures sont disponibles sur GitHub.. Les versions de build répertoriées dans le tableau ne sont pas spécifiées dans les versions autogérées répertoriées sur GitHub. Mettez à jour votre version en procédant comme suit, selon le cas :

   - Si vous n'avez aucun paramètre personnalisé pour le module complémentaire, exécutez la commande sous l'`To apply this release:`en-tête correspondant à la version vers laquelle vous souhaitez effectuer la mise à jour. GitHub
   - Si vous avez personnalisé des paramètres, téléchargez le fichier manifeste avec la commande suivante. *`Remplacez https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.2/config/master/aws-k8s-cni.yaml`* par l'URL de la version vers GitHub laquelle vous souhaitez effectuer la mise à jour.

     ```
     curl -O https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.2/config/master/aws-k8s-cni.yaml
     ```

     Si nécessaire, modifiez le manifeste avec les paramètres personnalisés de la sauvegarde que vous avez effectuée lors d'une étape précédente, puis appliquez le manifeste modifié à votre cluster. Si vos nœuds n'ont pas accès aux référentiels privés Amazon EKS Amazon ECR à partir desquels les images sont extraites (voir les lignes commençant par `image:` dans le manifeste), vous devrez télécharger les images, les copier dans votre propre référentiel et modifier le manifeste pour extraire les images de votre référentiel. Pour plus d'informations, consultez Copier une image de conteneur d'un référentiel vers un autre référentiel.

     ```
     kubectl apply -f aws-k8s-cni.yaml
     ```

5. Vérifiez que la nouvelle version est maintenant installée sur votre cluster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

L'exemple qui suit illustre un résultat.

```
v1.18.2
```

6. (Facultatif) Installez le `cni-metrics-helper` sur votre cluster. Il collecte les informations relatives à l'interface Elastic Network et aux adresses IP, les agrège au niveau du cluster et publie les statistiques sur Amazon. CloudWatch Pour plus d'informations, consultez cni-metrics-helper on. GitHub

Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA)

Le Amazon VPC CNI plugin for Kubernetes est le plugin de mise en réseau de la mise en réseau du Pod dans les clusters Amazon EKS. Le plugin est chargé d'allouer des adresses IP VPC aux nœuds Kuberneteset de configurer la mise en réseau nécessaire pour les Pods sur chaque nœud. Le plugin :

- Nécessite des autorisations AWS Identity and Access Management (IAM). Si votre cluster utilise la famille IPv4, les autorisations sont spécifiées dans la politique `AmazonEKS_CNI_Policy` AWS gérée. Si votre cluster utilise la famille IPv6, les autorisations doivent être ajoutées à une politique IAM que vous créez.Vous pouvez attacher cette politique au rôle IAM de nœud Amazon EKS ou à un rôle IAM distinct. Nous vous recommandons de l'affecter à un rôle séparé, comme détaillé dans cette rubrique.
- Crée et est configuré pour utiliser un compte de service Kubernetes nommé `aws-node` quand il est déployé. Le compte de service est lié à un `clusterrole` Kubernetes nommé `aws-node`, qui reçoit les autorisations Kubernetes requises.

> **Note**
>
> Les Pods pour Amazon VPC CNI plugin for Kubernetes ont accès aux autorisations attribuées au rôle IAM de nœud Amazon EKS, sauf si vous bloquez l'accès à IMDS. Pour plus d'informations, consultez Restreindre l'accès au profil d'instance affecté au composant master.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.
- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.

Étape 1 : création du rôle IAM Amazon VPC CNI plugin for Kubernetes

Pour créer le rôle IAM

1. Déterminez la famille d'adresses IP de votre cluster.

   ```
   aws eks describe-cluster --name my-cluster | grep ipFamily
   ```

   L'exemple qui suit illustre un résultat.

   ```
   "ipFamily": "ipv4"
   ```

   La sortie peut renvoyer `ipv6` à la place.

2. Créez le rôle IAM. Vous pouvez utiliser `eksctl` ou `kubectl` et le AWS CLI pour créer votre rôle IAM.

   eksctl

   Créez un rôle IAM et attachez la politique IAM au rôle avec la commande qui correspond à la famille d'adresses IP de votre cluster. La commande crée et déploie une AWS CloudFormation pile qui crée un rôle IAM, y attache la politique que vous spécifiez et annote le compte de `aws-node` Kubernetes service existant avec l'ARN du rôle IAM créé.

   - `IPv4`

     Remplacez *my-cluster* par votre propre valeur.

     ```
     eksctl create iamserviceaccount \
         --name aws-node \
         --namespace kube-system \
         --cluster my-cluster \
         --role-name AmazonEKSVPCCNIRole \
     ```

```
    --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --override-existing-serviceaccounts \
    --approve
```

- IPv6

  Remplacez *my-cluster* par votre propre valeur. Remplacez *111122223333* par l'ID de votre compte et remplacez *AmazonEKS_CNI_IPv6_Policy* par le nom de votre politique IPv6. Si vous ne disposez pas d'une politique IPv6, consultez Créer une politique IAM pour les clusters qui utilisent la famille IPv6 pour en créer une. Pour utiliser IPv6 avec votre cluster, ce dernier doit répondre à plusieurs exigences. Pour plus d'informations, consultez IPv6adresses pour les clustersPods, et services.

```
eksctl create iamserviceaccount \
    --name aws-node \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKSVPCCNIRole \
    --attach-policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
    --override-existing-serviceaccounts \
    --approve
```

kubectl and the AWS CLI

1. Affichez l'URL du fournisseur OIDC de votre cluster.

   ```
   aws eks describe-cluster --name my-cluster --query
    "cluster.identity.oidc.issuer" --output text
   ```

   L'exemple qui suit illustre un résultat.

   ```
   https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
   ```

   Si aucune sortie n'est renvoyée, vous devez créer un fournisseur OIDC IAM pour votre cluster.

2. Copiez le contenu suivant dans un fichier nommé *vpc-cni-trust-policy.json*. Remplacez *111122223333* par votre ID de compte et

EXAMPLED539D4633E53DE1B71EXAMPLE par la sortie renvoyée à l'étape précédente. *region-code*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-node"
                }
            }
        }
    ]
}
```

3. Créez le rôle. Vous pouvez remplacer *AmazonEKSVPCCNIRole* par n'importe quel nom que vous choisissez.

```
aws iam create-role \
  --role-name AmazonEKSVPCCNIRole \
  --assume-role-policy-document file://"vpc-cni-trust-policy.json"
```

4. Attachez la politique IAM requise au rôle. Exécutez la commande qui correspond à la famille d'adresses IP de votre cluster.

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSVPCCNIRole
     ```

   - IPv6

Remplacez *111122223333* par l'ID de votre compte et *AmazonEKS_CNI_IPv6_Policy* par le nom de votre politique IPv6. Si vous ne disposez pas d'une politique IPv6, consultez Créer une politique IAM pour les clusters qui utilisent la famille IPv6 pour en créer une. Pour utiliser IPv6 avec votre cluster, ce dernier doit répondre à plusieurs exigences. Pour plus d'informations, consultez IPv6adresses pour les clustersPods, et services.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSVPCCNIRole
```

5. Exécutez la commande suivante pour annoter le compte de service aws-node avec l'ARN du rôle IAM que vous avez créé précédemment. Remplacez les *example values* par vos propres valeurs.

```
kubectl annotate serviceaccount \
    -n kube-system aws-node \
    eks.amazonaws.com/role-
arn=arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

3. (Facultatif) Configurez le type de AWS Security Token Service point de terminaison utilisé par votre compte Kubernetes de service. Pour plus d'informations, consultez Configuration du AWS Security Token Service point de terminaison pour un compte de service.

Étape 2 : redéploiement des Pods du Amazon VPC CNI plugin for Kubernetes

1. Supprimez et recréez tous les Pods existants associés au compte de service pour appliquer les variables d'environnement d'informations d'identification. L'annotation n'est pas appliquée aux Pods qui sont actuellement en cours d'exécution sans l'annotation. La commande suivante supprime les Pods de aws-node DaemonSet existants et les déploie avec l'annotation de compte de service.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

2. Vérifiez que les Pods ont tous redémarré.

```
kubectl get pods -n kube-system -l k8s-app=aws-node
```

3. Décrivez l'un des Pods et vérifiez que les variables d'environnement `AWS_WEB_IDENTITY_TOKEN_FILE` et `AWS_ROLE_ARN` existent. Remplacez *cpjw7* par le nom de l'un de vos Pods renvoyés dans la sortie de l'étape précédente.

```
kubectl describe pod -n kube-system aws-node-cpjw7 | grep 'AWS_ROLE_ARN:\|
AWS_WEB_IDENTITY_TOKEN_FILE:'
```

L'exemple qui suit illustre un résultat.

```
AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
      AWS_ROLE_ARN:
 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:           /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

Deux ensembles de résultats dupliqués sont renvoyés, car le Pod comprend deux conteneurs. Les deux exemples présentent les mêmes valeurs.

Si vous utilisez Pod le point de terminaison Région AWS al, la ligne suivante est également renvoyée dans la sortie précédente.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Étape 3 : suppression de la politique CNI du rôle IAM de nœud

Si le rôle IAM de votre nœud Amazon EKS est actuellement associé à la politique `AmazonEKS_CNI_Policy` IAM (`IPv4`) ou à une `IPv6`politique, et que vous avez créé un rôle IAM distinct, que vous y avez attaché la politique et que vous l'avez attribué au compte de `aws-node` Kubernetes service, nous vous recommandons de supprimer la politique de votre rôle de nœud à l'aide de la AWS CLI commande correspondant à la famille d'adresses IP de votre cluster. Remplacez *AmazonEKSNodeRole* par le nom de votre rôle de nœud.

- `IPv4`

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
```

- IPv6

  Remplacez *111122223333* par l'ID de votre compte et *AmazonEKS_CNI_IPv6_Policy* par le nom de votre politique IPv6.

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy
```

Créer une politique IAM pour les clusters qui utilisent la famille **IPv6**

Si vous avez créé un cluster qui utilise la famille IPv6 et que la version 1.10.1 ou une version ultérieure du module complémentaire Amazon VPC CNI plugin for Kubernetes du cluster est configurée, vous devez créer une politique IAM que vous pourrez affecter à un rôle IAM. Si vous avez un cluster existant que vous n'avez pas configuré avec la famille IPv6 lorsque vous l'avez créé, alors pour utiliser IPv6, vous devez créer un nouveau cluster. Pour plus d'informations sur l'utilisation de IPv6 avec votre cluster, veuillez consulter IPv6adresses pour les clustersPods, et services.

1. Copiez la politique suivante et enregistrez-la dans un fichier appelé *vpc-cni-ipv6-policy*.json.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
```

```
            ]
        }
    ]
}
```

2. Créez la politique IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

Choix de cas d'utilisation de mise en réseau de Pod

Le Amazon VPC CNI plugin for Kubernetes fournit une mise en réseau pour Pods. Le tableau suivant vous aide à comprendre les cas d'utilisation de la mise en réseau que vous pouvez utiliser ensemble, ainsi que les fonctionnalités et les paramètres Amazon VPC CNI plugin for Kubernetes que vous pouvez utiliser avec différents types de nœuds Amazon EKS. Toutes les informations du tableau s'appliquent uniquement aux nœuds `IPv4` Linux.

Type de nœud Amazon EKS	Amazon EC2			Fargate
Cas d'utilisation	Adresses IP individuelles assignées à l'interface réseau	Préfixes IP assignés à l'interface réseau	Groupes de sécurité pour Pods	
Mise en réseau personnalisée pour les pods : affecte des adresses IP à partir d'un sous-réseau différent du sous-réseau du nœud	Oui	Oui	Oui	Oui (sous-réseaux contrôlés par le profil Fargate)

Type de nœud Amazon EKS	Amazon EC2			Fargate
Cas d'utilisation	Adresses IP individuelles assignées à l'interface réseau	Préfixes IP assignés à l'interface réseau	Groupes de sécurité pour Pods	
SNAT pour Pods	Oui (la valeur par défaut est `false`)	Oui (la valeur par défaut est `false`)	Oui (uniquement `true`)	Oui (uniquement `true`)
Capacités				
Portée du groupe de sécurité	Nœud	Nœud	Pod (si vous avez défini `POD_SECURITY_GROUP_ENFORCING_MODE = standard` et `AWS_VPC_K8S_CNI_EXTERNALSNAT =false`, le trafic destiné aux points de terminaison situés en dehors du VPC utilise les groupes de sécurité du nœud, et non les groupes de sécurité de Pod's)	Pod

Type de nœud Amazon EKS	Amazon EC2			Fargate
Cas d'utilisation	Adresses IP individuelles assignées à l'interface réseau	Préfixes IP assignés à l'interface réseau	Groupes de sécurité pour Pods	
Types de sous-réseau Amazon VPC	Publics et privés	Publics et privés	Privés uniquement	Privés uniquement
Stratégie réseau (VPC CNI)	Compatible	Compatible	Compatible Uniquement avec la version `1.14.0` ou ultérieure du plug-in Amazon VPC CNI	Non pris en charge
Densité de pod par nœud	Medium	Élevée	Faible	Un
Heure de lancement du pod	Mieux	Meilleure	Bon	Modérée
Paramètres du plug-in Amazon VPC CNI (pour plus d'informations sur chaque paramètre, voir amazon-vpc-cni-k 8s on) GitHub				
`WARM_ENI_TARGET`	Oui	Ne s'applique pas	Ne s'applique pas	Ne s'applique pas
`WARM_IP_TARGET`	Oui	Oui	Ne s'applique pas	Ne s'applique pas
`MINIMUM_IP_TARGET`	Oui	Oui	Ne s'applique pas	Ne s'applique pas

Type de nœud Amazon EKS	Amazon EC2			Fargate
Cas d'utilisation	Adresses IP individuelles assignées à l'interface réseau	Préfixes IP assignés à l'interface réseau	Groupes de sécurité pour Pods	
WARM_PREFIX_TARGET	Ne s'applique pas	Oui	Ne s'applique pas	Ne s'applique pas

> **Note**
>
> - Vous ne pouvez pas utiliser IPv6 avec une mise en réseau personnalisée.
> - Les adresses IPv6 ne sont pas traduites, le SNAT ne s'applique donc pas.
> - Le flux de trafic vers et depuis les Pods avec des groupes de sécurité associés n'est pas soumis à l'application de la stratégie de réseau Calico et est limité à l'application du groupe de sécurité Amazon VPC.
> - Si vous appliquez une politique Calico réseau, nous vous recommandons de définir la variable d'environnement ANNOTATE_POD_IP sur `true` afin d'éviter un problème connu avecKubernetes. Pour utiliser cette fonctionnalité, vous devez ajouter une `patch` autorisation pour les modules au `aws-node`ClusterRole. Notez que l'ajout d'autorisations de correctif `aws-node` DaemonSet augmente la portée de sécurité du plugin. Pour plus d'informations, consultez ANNOTATE_POD_IP dans le référentiel VPC CNI sur. GitHub
> - Les préfixes IP et les adresses IP sont associés aux interfaces réseau Elastic Amazon EC2 standards. Les pods nécessitant des groupes de sécurité spécifiques se voient attribuer l'adresse IP principale d'une interface réseau de branche. Vous pouvez associer des Pods qui obtiennent des adresses IP ou des adresses IP à partir de préfixes IP avec des Pods qui obtiennent des interfaces réseau de branche sur le même nœud.

Nœuds Windows

Chaque nœud ne prend en charge qu'une seule interface réseau. Vous pouvez utiliser des adresses IPv4 et des préfixes IPv4 secondaires. Par défaut, le nombre d'adresses IPv4 disponibles sur

le nœud est égal au nombre d'adresses IPv4 secondaires que vous pouvez attribuer à chaque interface réseau élastique, moins une. Toutefois, vous pouvez augmenter les adresses IPv4 disponibles et la densitéPod sur le nœud en activant les préfixes IP. Pour plus d'informations, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.

Les stratégies réseau Calico sont prises en charge sous Windows. Vous ne pouvez pas utiliser les groupes de sécurité pour Pods ou la mise en réseau personnalisée sous Windows.

IPv6adresses pour les clustersPods, et services

Par défaut, Kubernetes attribue des adresses IPv4 à vos Pods et services. Au lieu d'attribuer des adresses IPv4 à vos Pods et services, vous pouvez configurer votre cluster pour leur attribuer des adresses IPv6. Amazon EKS ne prend pas en charge le double empilement Pods ou services, bien que Kubernetes le prenne en charge dans sa version1.23 et dans ses version ultérieures. Par conséquent, vous ne pouvez pas attribuer à la fois des adresses IPv4 et IPv6 à vos Pods et services.

Vous sélectionnez la famille IP que vous souhaitez utiliser pour votre cluster lorsque vous le créez. Vous ne pouvez pas changer la famille après avoir créé le cluster.

Considérations relatives à l'utilisation de la **IPv6** famille pour votre cluster

- Vous devez créer un nouveau cluster et spécifier que vous voulez utiliser la famille IPv6 pour ce cluster. Vous ne pouvez pas activer la famille IPv6 pour un cluster que vous avez mis à jour à partir d'une version précédente. Pour obtenir des instructions sur la création d'un cluster, veuillez consulter Création d'un cluster Amazon EKS.
- La version du module complémentaire CNI Amazon VPC que vous déployez sur votre cluster doit être la version 1.10.1 ou ultérieure. Cette version ou une version ultérieure est déployée par défaut. Après avoir déployé le module complémentaire, vous ne pouvez pas rétrograder votre module complémentaire CNI Amazon VPC vers une version inférieure à 1.10.1 sans supprimer au préalable tous les nœuds de tous les groupes de nœuds de votre cluster.
- Windows Pods et services ne sont pas pris en charge.
- Si vous utilisez des nœuds Amazon EC2, vous devez configurer le module complémentaire CNI Amazon VPC pour qu'il utilise IPv6 et la délégation de préfixe IP. Si vous choisissez la famille IPv6 lors de la création de votre cluster, la version 1.10.1 du module complémentaire correspond par défaut à cette configuration. C'est le cas pour un module complémentaire auto-géré ou Amazon EKS. Pour de plus amples informations sur la délégation de préfixes IP, veuillez consulter Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2.

- Lorsque vous créez un cluster, le VPC et les sous-réseaux que vous spécifiez doivent posséder un bloc d'adresse CIDR IPv6 affecté au VPC et aux sous-réseaux que vous spécifiez. Ils doivent également avoir un bloc d'adresse CIDR IPv4 qui leur est affecté. En effet, même si vous souhaitez uniquement utiliser IPv6, un VPC nécessite toujours un bloc d'adresse CIDR IPv4 pour fonctionner. Pour plus d'informations, veuillez consulter la section Associer un bloc d'adresse CIDR IPv6 à votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Lorsque vous créez votre cluster et vos nœuds, vous devez spécifier des sous-réseaux configurés pour attribuer automatiquement des adresses IPv6. Sinon, vous ne pourrez pas déployer votre cluster et vos nœuds. Par défaut, cette configuration est désactivée. Pour plus d'informations, consultez Modification de l'attribut d'adressage IPv6 de votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.
- Les tables de routage affectées à vos sous-réseaux doivent comporter des routes pour les adresses IPv6. Pour de plus amples informations, veuillez consulter la section Migrer vers IPv6 du Guide de l'utilisateur Amazon VPC.
- Vos groupes de sécurité doivent autoriser les adresses IPv6. Pour de plus amples informations, veuillez consulter la section Migrer vers IPv6 du Guide de l'utilisateur Amazon VPC.
- Vous ne pouvez l'utiliser qu'IPv6avec des nœuds Amazon EC2 ou Fargate AWS basés sur Nitro.
- Vous ne pouvez pas utiliser IPv6 avec Groupes de sécurité pour Pods sur des nœuds Amazon EC2. Toutefois, vous pouvez l'utiliser avec des nœuds Fargate. Si vous avez besoin de groupes de sécurité distincts pour des Pods individuels, continuez à utiliser la famille IPv4 avec des nœuds Amazon EC2 ou utilisez plutôt des nœuds Fargate.
- Si vous avez déjà utilisé la mise en réseau personnalisée pour atténuer l'épuisement des adresses IP, vous pouvez utiliser IPv6 à la place. Vous ne pouvez pas utiliser IPv6 avec une mise en réseau personnalisée. Si vous utilisez un réseau personnalisé pour l'isolation du réseau, vous devrez peut-être continuer à utiliser un réseau personnalisé et la famille IPv4 pour vos clusters.
- Vous ne pouvez pas utiliser IPv6 avec AWS Outposts.
- Les Pods et les services se voient attribuer uniquement une adresse IPv6. Aucune adresse IPv4 ne leur est affectée. Les Pods étant capables de communiquer avec des points de terminaison IPv4 par le biais de la NAT sur l'instance elle-même, DNS64 et NAT64 ne sont pas nécessaires. Si le trafic nécessite une adresse IP publique, l'adresse du réseau source est alors traduite en une IP publique.
- L'adresse IPv6 source d'un Pod n'est pas traduite en adresse IPv6 du nœud lorsqu'il communique en dehors du VPC. Il est acheminé par une passerelle Internet ou une passerelle Internet de sortie uniquement.

- Tous les nœuds se voient attribuer une adresse `IPv4` et `IPv6`.
- Le Pilote CSI Amazon FSx pour Lustre n'est pas pris en charge.
- Vous pouvez utiliser la version `2.3.1` ou une version ultérieure du AWS Load Balancer Controller pour équilibrer la charge du trafic des applications ou du réseau `IPv6` Pods en mode IP, mais pas en mode instance. Pour plus d'informations, consultez Qu'est-ce que AWS Load Balancer Controller ?.
- Vous devez attacher une politique `IPv6` IAM à votre rôle IAM de nœud ou CNI IAM. Entre les deux, nous vous recommandons de le rattacher à un rôle CNI IAM. Pour plus d'informations, consultez Créer une politique IAM pour les clusters qui utilisent la famille `IPv6` et Étape 1 : création du rôle IAM Amazon VPC CNI plugin for Kubernetes.
- Chaque Pod Fargate reçoit une adresse `IPv6` à partir du CIDR spécifié pour le sous-réseau dans lequel il est déployé. L'unité matérielle sous-jacente qui exécute des Pods Fargate obtient une adresse `IPv4` et `IPv6` uniques à partir des CIDR affectés au sous-réseau dans lequel l'unité matérielle est déployée.
- Nous vous recommandons d'effectuer une évaluation approfondie de vos applications, des modules complémentaires Amazon EKS et AWS des services auxquels vous intégrez avant de déployer des `IPv6` clusters. Cela permet de s'assurer que tout fonctionne comme prévu avec `IPv6`.
- L'utilisation du point de terminaison `IPv6` du service de métadonnées d'instance Amazon EC2 n'est pas prise en charge avec Amazon EKS.
- Lors de la création d'un groupe de nœuds autogéré dans un cluster qui utilise la famille `IPv6`, les données utilisateur doivent inclure les `BootstrapArguments` suivants pour le fichier `bootstrap.sh` qui s'exécute au démarrage du nœud. Remplacez *`your-cidr-`* par la portée `IPv6` CIDR de l'ensemble de cluster de votre VPC.

```
--ip-family ipv6 --service-ipv6-cidr your-cidr
```

Si vous ne connaissez pas la `IPv6` `CIDR` plage de votre cluster, vous pouvez la voir à l'aide de la commande suivante (nécessite la AWS CLI version `2.4.9` ou une version ultérieure).

```
aws eks describe-cluster --name my-cluster --query
 cluster.kubernetesNetworkConfig.serviceIpv6Cidr --output text
```

Déploiement d'un cluster **IPv6** et de nœuds gérés par Amazon Linux

Dans ce ditacticiel, vous déployez un Amazon VPC IPv6, un cluster Amazon EKS avec la famille IPv6 et un groupe de nœuds gérés avec des nœuds Amazon EC2 Amazon Linux. Vous ne pouvez pas déployer des nœuds Amazon EC2 Windows dans un cluster IPv6. Vous pouvez également déployer des nœuds Fargate dans votre cluster, mais ces instructions ne sont pas fournies dans cette rubrique pour des raisons de simplicité.

Avant de créer un cluster pour une utilisation en production, nous vous recommandons de vous familiariser avec tous les paramètres et de déployer un cluster avec les paramètres qui répondent à vos besoins. Pour de plus amples informations, veuillez consulter Création d'un cluster Amazon EKS, Groupes de nœuds gérés et les considérations relatives ce sujet. Vous ne pouvez activer certains paramètres que lors de la création de votre cluster.

Prérequis

Avant de démarrer ce didacticiel, vous devez installer et configurer les outils et les ressources suivants dont vous avez besoin pour créer et gérer un cluster Amazon EKS.

- L'outil de ligne de commande kubectl est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est 1.29, vous pouvez utiliser la version kubectl 1.28, 1.29 ou 1.30. Pour installer ou mettre à niveau kubectl, veuillez consulter Installation ou mise à jour de kubectl.
- Le principal de sécurité IAM que vous utilisez doit être autorisé à utiliser les rôles IAM Amazon EKS, les rôles liés à un service AWS CloudFormation, un VPC et les ressources associées. Pour plus d'informations, consultez Actions, ressources et clés de condition pour Amazon Elastic Kubernetes Service et Utilisation des rôles liés à un service dans le guide de l'utilisateur IAM.

Des procédures sont fournies pour créer les ressources en utilisant soit eksctl, soit la AWS CLI. Vous pouvez également déployer les ressources à l'aide du AWS Management Console, mais ces instructions ne sont pas fournies dans cette rubrique pour des raisons de simplicité.

eksctl

Prérequis

Version 0.183.0 ou ultérieure de eksctl installée sur votre ordinateur. Pour l'installer ou le mettre à jour, veuillez consulter Installation dans la documentation de eksctl.

Pour déployer un cluster **IPv6** avec **eksctl**

1. Créez le fichier `ipv6-cluster.yaml`. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée :

   - Remplacez *my-cluster* par un nom pour votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
   - Remplacez *region-code* par n'importe quelle Région AWS prise en charge par Amazon EKS. Pour en obtenir la liste Régions AWS, consultez la section Points de terminaison et quotas Amazon EKS dans le guide de référence AWS général.
   - La valeur de `version` correspond à la version de votre cluster. Pour obtenir plus d'informations, consultez la version de Kubernetes prise en charge par Amazon EKS.
   - Remplacer *my-nodegroup* avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.
   - Remplacez *t3.medium* par tout type d'instance du système AWS Nitro.

```
cat >ipv6-cluster.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "X.XX"

kubernetesNetworkConfig:
  ipFamily: IPv6

addons:
  - name: vpc-cni
    version: latest
  - name: coredns
```

```
    version: latest
  - name: kube-proxy
    version: latest

iam:
  withOIDC: true

managedNodeGroups:
  - name: my-nodegroup
    instanceType: t3.medium
EOF
```

2. Créer votre cluster.

```
eksctl create cluster -f ipv6-cluster.yaml
```

La création du cluster prend plusieurs minutes. Ne continuez pas avant d'avoir vu la dernière ligne de sortie, qui ressemble à la sortie suivante.

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

3. Vérifiez que les adresses IPv6 sont attribuées aux Pods par défaut.

```
kubectl get pods -n kube-system -o wide
```

L'exemple qui suit illustre un résultat.

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

```
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

4. Vérifiez que les adresses IPv6 sont attribuées aux services par défaut.

   ```
   kubectl get services -n kube-system -o wide
   ```

   L'exemple qui suit illustre un résultat.

   ```
   NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
    SELECTOR
   kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
    k8s-app=kube-dns
   ```

5. (Facultatif) Déployez un exemple d'application ou déployez le AWS Load Balancer Controller et un exemple d'application pour équilibrer la charge des applications ou du trafic réseau vers les IPv6 Pods.
6. Une fois que vous avez terminé avec le cluster et les nœuds que vous avez créés pour ce tutoriel, vous devez nettoyer les ressources que vous avez créées avec la commande suivante.

   ```
   eksctl delete cluster my-cluster
   ```

AWS CLI

Prérequis

Version 2.12.3 ou ultérieure ou version 1.27.160 ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **aws --version | cut -d / -f2 | cut -d ' ' -f1**. Les gestionnaires de package, par exemple yum, apt-get, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez

Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur. Si vous utilisez le AWS CloudShell, vous devrez peut-être installer la version `2.12.3` ou une version ultérieure du AWS CLI, car la AWS CLI version par défaut installée dans le AWS CloudShell peut être une version antérieure. `1.27.160`

⚠ Important

- Vous devez effectuer toutes les étapes de cette procédure avec le même utilisateur. Exécutez la commande suivante pour vérifier l'utilisateur actuel :

  ```
  aws sts get-caller-identity
  ```

- Vous devez effectuer toutes les étapes de cette procédure dans le même shell. Plusieurs étapes utilisent des variables définies dans les étapes précédentes. Les étapes qui utilisent des variables ne fonctionneront pas correctement si les valeurs de ces variables sont définies dans un autre shell. Si vous utilisez le plugin AWS CloudShell pour effectuer la procédure suivante, n'oubliez pas que si vous n'interagissez pas avec celui-ci à l'aide de votre clavier ou de votre pointeur pendant environ 20 à 30 minutes, votre session shell se termine. Les processus en cours d'exécution ne sont pas considérés comme des interactions.
- Les instructions sont écrites pour le shell Bash et peuvent nécessiter des ajustements dans d'autres shells.

Pour créer votre cluster à l'aide du AWS CLI

Remplacez toutes les *`example values`* dans les étapes de cette procédure par vos propres valeurs.

1. Exécutez les commandes suivantes pour définir certaines variables utilisées dans les étapes suivantes. *`region-code`*Remplacez-le par Région AWS celui dans lequel vous souhaitez déployer vos ressources. La valeur peut être n'importe Région AWS laquelle prise en charge par Amazon EKS. Pour en obtenir la liste Régions AWS, consultez la section Points de terminaison et quotas Amazon EKS dans le guide de référence AWS général.

Remplacez *my-cluster* par un nom pour votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Remplacer *my-nodegroup* avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Remplacez *111122223333* par votre ID de compte.

```
export region_code=region-code
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
```

2. Créez un Amazon VPC avec des sous-réseaux publics et privés qui répondent aux exigences Amazon EKS et IPv6.

   a. Exécutez la commande suivante pour définir une variable pour le nom de votre AWS CloudFormation pile. Vous pouvez remplacer *my-eks-ipv6-vpc* par n'importe quel nom que vous choisissez.

   ```
   export vpc_stack_name=my-eks-ipv6-vpc
   ```

   b. Créez un IPv6 VPC à l'aide d'un AWS CloudFormation modèle.

   ```
   aws cloudformation create-stack --region $region_code --stack-name
    $vpc_stack_name \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-
   subnets.yaml
   ```

   La création de la pile prend quelques minutes. Exécutez la commande suivante. Ne passez pas à l'étape suivante tant que la sortie de la commande n'est pas CREATE_COMPLETE.

   ```
   aws cloudformation describe-stacks --region $region_code --stack-name
    $vpc_stack_name --query Stacks[].StackStatus --output text
   ```

   c. Récupérez les ID des sous-réseaux publics qui ont été créés.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
output text
```

L'exemple qui suit illustre un résultat.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE
```

d. Activez l'option d'attribution automatique d'adresses IPv6 pour les sous-réseaux publics qui ont été créés.

```
aws ec2 modify-subnet-attribute --region $region_code --
subnet-id subnet-0a1a56c486EXAMPLE --assign-ipv6-address-on-
creation
aws ec2 modify-subnet-attribute --region $region_code --subnet-id
 subnet-099e6ca77aEXAMPLE --assign-ipv6-address-on-creation
```

e. Récupérez les noms des sous-réseaux et des groupes de sécurité créés par le modèle à partir de la AWS CloudFormation pile déployée et stockez-les dans des variables pour une utilisation ultérieure.

```
security_groups=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SecurityGroups`].OutputValue' --
output text)

public_subnets=$(aws cloudformation describe-stacks --region $region_code --
stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
output text)

private_subnets=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPrivate`].OutputValue' --
output text)

subnets=${public_subnets},${private_subnets}
```

3. Créez un rôle IAM de cluster et associez-y la politique gérée Amazon EKS IAM requise. Kubernetesles clusters gérés par Amazon EKS appellent d'autres AWS services en votre nom pour gérer les ressources que vous utilisez avec le service.

   a. Exécutez la commande suivante pour créer un fichier `eks-cluster-role-trust-policy.json` :

   ```
   cat >eks-cluster-role-trust-policy.json <<EOF
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "eks.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   EOF
   ```

   b. Exécutez la commande suivante pour définir une variable pour votre nom de rôle. Vous pouvez remplacer *myAmazonEKSClusterRole* par n'importe quel nom que vous choisissez.

   ```
   export cluster_role_name=myAmazonEKSClusterRole
   ```

   c. Créez le rôle.

   ```
   aws iam create-role --role-name $cluster_role_name --assume-role-policy-
   document file://"eks-cluster-role-trust-policy.json"
   ```

   d. Récupérez l'ARN du rôle IAM et le stockez dans une variable pour une étape ultérieure.

   ```
   cluster_iam_role=$(aws iam get-role --role-name $cluster_role_name --
   query="Role.Arn" --output text)
   ```

   e. Attachez la politique IAM gérée par Amazon EKS au rôle.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name $cluster_role_name
```

4. Créer votre cluster.

```
aws eks create-cluster --region $region_code --name $cluster_name --kubernetes-
version 1.XX \
   --role-arn $cluster_iam_role --resources-vpc-config subnetIds=
$subnets,securityGroupIds=$security_groups \
   --kubernetes-network-config ipFamily=ipv6
```

-
> **Note**
>
> Il est possible que vous receviez un message d'erreur indiquant que l'une des zones de disponibilité de votre demande ne dispose pas d'une capacité suffisante pour créer un cluster Amazon EKS. Si cela se produit, la sortie de l'erreur contient les zones de disponibilité qui peuvent prendre en charge un nouveau cluster. Essayez à nouveau de créer votre cluster avec au moins deux sous-réseaux situés dans les zones de disponibilité prises en charge pour votre compte. Pour plus d'informations, consultez Capacité insuffisante.

La création du cluster prend quelques minutes. Exécutez la commande suivante. Ne passez pas à l'étape suivante tant que la sortie de la commande n'est pas `ACTIVE`.

```
aws eks describe-cluster --region $region_code --name $cluster_name --query
 cluster.status
```

5. Créez ou mettez à jour un fichier `kubeconfig` pour votre cluster afin de pouvoir communiquer avec votre cluster.

```
aws eks update-kubeconfig --region $region_code --name $cluster_name
```

Par défaut, le paramètre `config` est créé dans `~/.kube` ou la configuration du nouveau cluster est ajoutée à un `config` fichier dans `~/.kube`.

6. Créez un rôle IAM de nœud.

   a. Exécutez la commande suivante pour créer un fichier `vpc-cni-ipv6-policy.json` :

```
cat >vpc-cni-ipv6-policy <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
EOF
```

b. Créez la politique IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

c. Exécutez la commande suivante pour créer un fichier `node-role-trust-relationship.json` :

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

d. Exécutez la commande suivante pour définir une variable pour votre nom de rôle. Vous pouvez remplacer *AmazonEKSNodeRole* par n'importe quel nom que vous choisissez.

```
export node_role_name=AmazonEKSNodeRole
```

e. Créez le rôle IAM.

```
aws iam create-role --role-name $node_role_name --assume-role-policy-
document file://"node-role-trust-relationship.json"
```

f. Attachez la politique IAM au rôle IAM.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Pour simplifier ce tutoriel, la politique est attachée à ce rôle IAM. Toutefois, dans un cluster de production, nous recommandons de rattacher la politique à un rôle IAM distinct. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

g. Attachez deux politiques gérées IAM requises au rôle IAM.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly \
```

```
  --role-name $node_role_name
```

h. Récupérez l'ARN du rôle IAM et le stockez dans une variable pour une étape ultérieure.

```
node_iam_role=$(aws iam get-role --role-name $node_role_name --
query="Role.Arn" --output text)
```

7. Créez un groupe de nœuds gérés.

a. Affichez les ID des sous-réseaux que vous avez créés à l'étape précédente.

```
echo $subnets
```

L'exemple qui suit illustre un résultat.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE,subnet-
0377963d69EXAMPLE,subnet-0c05f819d5EXAMPLE
```

b. Créez le groupe de nœuds. Remplacez *0a1a56c486EXAMPLE*, *099e6ca77aEXAMPLE*, *0377963d69EXAMPLE*, et *0c05f819d5EXAMPLE* par les valeurs renvoyées dans la sortie de l'étape précédente. Veillez à supprimer les virgules entre les ID de sous-réseau de la sortie précédente dans la commande suivante. Vous pouvez remplacer *t3.medium* par tout type d'instance du système AWS Nitro.

```
aws eks create-nodegroup --region $region_code --cluster-name $cluster_name
 --nodegroup-name $nodegroup_name \
    --subnets subnet-0a1a56c486EXAMPLE subnet-099e6ca77aEXAMPLE
 subnet-0377963d69EXAMPLE subnet-0c05f819d5EXAMPLE \
    --instance-types t3.medium --node-role $node_iam_role
```

La création du groupe de nœuds prend quelques minutes. Exécutez la commande suivante. Ne passez pas à l'étape suivante tant que la sortie retournée n'est pas `ACTIVE`.

```
aws eks describe-nodegroup --region $region_code --cluster-name
 $cluster_name --nodegroup-name $nodegroup_name \
    --query nodegroup.status --output text
```

8. Vérifiez que les adresses `IPv6` sont attribuées aux Pods par défaut dans la colonne `IP`.

```
kubectl get pods -n kube-system -o wide
```

L'exemple qui suit illustre un résultat.

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

9. Vérifiez que les adresses IPv6 sont attribuées aux services par défaut dans la colonne IP.

```
kubectl get services -n kube-system -o wide
```

L'exemple qui suit illustre un résultat.

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
 k8s-app=kube-dns
```

10. (Facultatif) Déployez un exemple d'application ou déployez le AWS Load Balancer Controller et un exemple d'application pour équilibrer la charge des applications ou du trafic réseau vers les IPv6 Pods.

11. Une fois que vous avez terminé avec le cluster et les nœuds que vous avez créés pour ce tutoriel, vous devez nettoyer les ressources que vous avez créées avec les commandes suivantes. Assurez-vous que vous n'utilisez aucune des ressources en dehors de ce tutoriel avant de les supprimer.

    a. Si vous effectuez cette étape dans un shell différent de celui dans lequel vous avez effectué les étapes précédentes, définissez les valeurs de toutes les variables utilisées dans les étapes précédentes, en remplaçant les *example values* par les valeurs que vous avez spécifiées lorsque vous avez effectué les étapes précédentes. Si vous effectuez cette étape dans le même shell que celui dans lequel vous avez effectué les étapes précédentes, passez à l'étape suivante.

    ```
    export region_code=region-code
    export vpc_stack_name=my-eks-ipv6-vpc
    export cluster_name=my-cluster
    export nodegroup_name=my-nodegroup
    export account_id=111122223333
    export node_role_name=AmazonEKSNodeRole
    export cluster_role_name=myAmazonEKSClusterRole
    ```

    b. Supprimez votre groupe de nœuds.

    ```
    aws eks delete-nodegroup --region $region_code --cluster-name $cluster_name
     --nodegroup-name $nodegroup_name
    ```

    La suppression prend quelques minutes. Exécutez la commande suivante. Ne pas passer à l'étape suivante si une sortie est retournée.

    ```
    aws eks list-nodegroups --region $region_code --cluster-name $cluster_name
     --query nodegroups --output text
    ```

    c. Supprimez le cluster.

    ```
    aws eks delete-cluster --region $region_code --name $cluster_name
    ```

    La suppression du cluster prend quelques minutes. Avant de continuer, assurez-vous que le cluster est supprimé avec la commande suivante.

    ```
    aws eks describe-cluster --region $region_code --name $cluster_name
    ```

Ne passez pas à l'étape suivante tant que votre résultat n'est pas similaire au résultat suivant.

```
An error occurred (ResourceNotFoundException) when calling the
 DescribeCluster operation: No cluster found for name: my-cluster.
```

d. Supprimez les ressources IAM que vous avez créées. Remplacez *AmazonEKS_CNI_IPv6_Policy* par le nom que vous avez choisi, si vous avez choisi un nom différent de celui utilisé dans les étapes précédentes.

```
aws iam detach-role-policy --role-name $cluster_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-role --role-name $cluster_role_name
aws iam delete-role --role-name $node_role_name
```

e. Supprimez la AWS CloudFormation pile qui a créé le VPC.

```
aws cloudformation delete-stack --region $region_code --stack-name
 $vpc_stack_name
```

SNAT pour Pods

Si vous avez déployé votre cluster à l'aide de la famille `IPv6`, les informations de cette rubrique ne s'appliquent pas à votre cluster, car les adresses `IPv6` ne sont pas traduites en réseau. Pour plus d'informations sur l'utilisation des adresses `IPv6` avec votre cluster, veuillez consulter IPv6adresses pour les clustersPods, et services.

Par défaut, chaque Pod dans votre cluster se voit attribuer une adresse `IPv4` privée d'un bloc de routage inter-domaines sans classe (CIDR) associé au VPC dans lequel le Pod est déployé. Pods dans le même VPC communiquent entre eux en utilisant ces adresses IP privées comme points de terminaison. Lorsqu'un Pod communique avec n'importe quelle adresse `IPv4` qui ne se trouve pas

dans un bloc CIDR associé à votre VPC, le plug-in CNI Amazon VPC (pour Linux ou Windows) traduit l'adresse `IPv4` des Pod's en une adresse `IPv4` privée primaire de l'interface réseau Elastic primaire du nœud sur lequel le Pod s'exécute, par défaut `-*`.

> **Note**
>
> Pour les nœuds Windows, des détails supplémentaires doivent être pris en compte. Par défaut, le plug-in CNI VPC pour Windows est défini avec une configuration réseau dans laquelle le trafic vers une destination au sein du même VPC est exclu pour la SNAT. Cela signifie que la SNAT est désactivée pour la communication interne du VPC et que l'adresse IP attribuée à un Pod est acheminable au sein du VPC. Mais le trafic vers une destination en dehors du VPC fait en sorte que l'IP du Pod source soit traduite par SNAT vers l'adresse IP principale de l'ENI de l'instance. Cette configuration par défaut pour Windows garantit que le pod peut accéder à des réseaux extérieurs à votre VPC de la même manière que l'instance hôte.

Compte tenu de ce comportement :

- Vos Pods peuvent communiquer avec les ressources Internet uniquement si le nœud sur lequel ils s'exécutent s'est vu attribuer une adresse IP publique ou Elastic et se trouve dans un sous-réseau public. Une table de routage associée d'un sous-réseau public comporte une route vers une passerelle Internet. Nous recommandons de déployer les nœuds dans des sous-réseaux privés, dans la mesure du possible.
- Pour les versions du plug-in antérieures à la version `1.8.0`, les ressources se trouvant sur des réseaux ou des VPC connectés au VPC de votre cluster à l'aide de l'appairage de VPC, d'un VPC de transit ou d'AWS Direct Connect ne peuvent pas initier la communication avec vos Pods derrière des interfaces réseau Elastic secondaires. Vos Pods peuvent toutefois initier une communication avec ces ressources et recevoir des réponses de leur part.

Si l'une des affirmations suivantes est vraie dans votre environnement, modifiez la configuration par défaut à l'aide de la commande suivante.

- Vous disposez de ressources sur des réseaux ou des VPC connectés au VPC de votre cluster à l'aide de l'appairage de VPC, d'un VPC de transit ou d'AWS Direct Connect qui doivent initier une communication avec vos Pods à l'aide d'une adresse `IPv4` et la version de votre plug-in est antérieure à la version `1.8.0`.

- Vos Pods se trouvent dans un sous-réseau privé et doivent communiquer en sortie vers Internet. Le sous-réseau dispose d'une route vers une passerelle NAT.

```
kubectl set env daemonset -n kube-system aws-node AWS_VPC_K8S_CNI_EXTERNALSNAT=true
```

> **Note**
>
> Les variables de configuration CNI `AWS_VPC_K8S_CNI_EXTERNALSNAT` et `AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS` ne sont pas applicables aux nœuds Windows. La désactivation de la SNAT n'est pas prise en charge pour Windows. Quant à l'exclusion d'une liste de CIDR `IPv4` de la SNAT, vous pouvez la définir en spécifiant le paramètre `ExcludedSnatCIDRs` dans le script d'amorçage Windows. Pour plus d'informations sur ce paramètre, veuillez consulter la rubrique Paramètres de configuration du script d'amorçage.

*Si une spécification des Pod's contient `hostNetwork=true` (la valeur par défaut est `false`), son adresse IP n'est pas traduite vers une autre adresse. C'est le cas pour les Pods `kube-proxy` et Amazon VPC CNI plugin for Kubernetes qui s'exécutent sur votre cluster, par défaut. Pour ces Pods, l'adresse IP est identique à l'adresse IP principale du nœud. L'adresse IP des Pod's n'est pas traduite. Pour plus d'informations sur un Pod's `hostNetwork` paramètre, voir PodSpec v1 core dans la référence de l'KubernetesAPI.

Configurez votre cluster pour les stratégies réseau Kubernetes

Par défaut, il n'existe aucune restriction dans Kubernetes pour les adresses IP, les ports ou les connexions entre les Pods de votre cluster ou entre vos Pods et les ressources de tout autre réseau. Vous pouvez utiliser une stratégie réseau Kubernetes pour limiter le trafic réseau à destination et en provenance de vos Pods. Pour plus d'informations, consultez Stratégies réseau dans la documentation Kubernetes.

Si vous avez une version `1.13` ou antérieure du Amazon VPC CNI plugin for Kubernetes sur votre cluster, vous devez implémenter une solution tierce pour appliquer des stratégies réseau Kubernetes à votre cluster. La version `1.14` ou ultérieure du plug-in peut implémenter des politiques réseau, vous n'avez donc pas besoin d'utiliser une solution tierce. Dans cette rubrique, vous apprendrez à configurer votre cluster pour utiliser une stratégie réseau Kubernetes sur votre cluster sans utiliser de module complémentaire tiers.

Les stratégies réseau dans le Amazon VPC CNI plugin for Kubernetes sont prises en charge dans les configurations suivantes.

- Clusters Amazon EKS de version `1.25` et ultérieure.
- Version 1.14 ou ultérieure du Amazon VPC CNI plugin for Kubernetes sur votre cluster.
- Cluster configuré pour les adresses `IPv4` ou `IPv6`.
- Vous pouvez utiliser les stratégies réseau avec des groupes de sécurité pour les Pods. Grâce aux stratégies réseau, vous pouvez contrôler toutes les communications au sein du cluster. Avec les groupes de sécurité pourPods, vous pouvez contrôler l'accès Services AWS depuis les applications d'unPod.
- Vous pouvez utiliser les stratégies réseau avec mise en réseau personnalisée et délégation de préfixes.

Considérations

- Lorsque vous appliquez des stratégies réseau du Amazon VPC CNI plugin for Kubernetes pour votre cluster à l'aide du Amazon VPC CNI plugin for Kubernetes, vous pouvez appliquer les stratégies aux nœuds Linux Amazon EC2 uniquement. Vous ne pouvez pas appliquer les stratégies aux nœuds Fargate ou Windows.
- Si votre cluster utilise actuellement une solution tierce pour gérer les stratégies réseau Kubernetes, vous pouvez utiliser ces mêmes politiques avec le Amazon VPC CNI plugin for Kubernetes. Vous devez toutefois supprimer votre solution existante afin qu'elle ne gère pas les mêmes stratégies.
- Vous pouvez appliquer plusieurs stratégies réseau à un même Pod. Lorsque deux stratégies ou plus sélectionnent le même Pod sont configurés, toutes les stratégies sont appliquées au Pod.
- Le nombre maximum de combinaisons uniques de ports pour chaque protocole dans chaque `egress:` sélecteur `ingress:` d'une politique réseau est de 24.
- Pour n'importe lequel de vos services Kubernetes, le port de service doit être le même que le port de conteneur. Si vous utilisez des ports nommés, utilisez également le même nom dans les spécifications du service.
- Application des politiques au Pod démarrage

  Le Amazon VPC CNI plugin for Kubernetes configure les stratégies réseau pour les pods en parallèle avec le provisionnement des pods. Jusqu'à ce que toutes les politiques soient configurées pour le nouveau pod, les conteneurs du nouveau pod commenceront par une politique d'autorisation par défaut. C'est ce qu'on appelle le mode standard. Une politique d'autorisation par

défaut signifie que tout le trafic entrant et sortant est autorisé à destination et en provenance des nouveaux modules.

Vous pouvez modifier cette politique réseau par défaut en définissant la variable `NETWORK_POLICY_ENFORCING_MODE` d'environnement sur `strict` dans le `aws-node` conteneur du VPC CNI. `DaemonSet`

```
env:
  - name: NETWORK_POLICY_ENFORCING_MODE
    value: "strict"
```

Lorsque la `NETWORK_POLICY_ENFORCING_MODE` variable est définie sur`strict`, les pods qui utilisent le VPC CNI commencent par une politique de refus par défaut, puis les politiques sont configurées. C'est ce qu'on appelle le mode strict. En mode strict, vous devez disposer d'une politique réseau pour chaque point de terminaison auquel vos pods doivent accéder dans votre cluster. Notez que cette exigence s'applique aux CoreDNS capsules. La politique de refus par défaut n'est pas configurée pour les pods dotés d'un réseau hôte.

- La fonctionnalité de stratégie réseau crée et nécessite une définition de ressource personnalisée (CRD) de `PolicyEndpoint` appelée `policyendpoints.networking.k8s.aws`. Les objets du `PolicyEndpoint` de la ressource personnalisée sont gérés par Amazon EKS. Vous ne devez ni modifier ni supprimer ces ressources.
- Si vous exécutez des pods qui utilisent les informations d'identification IAM du rôle d'instance ou si vous vous connectez à l'IMDS EC2, veillez à vérifier les stratégies réseau susceptibles de bloquer l'accès à l'IMDS EC2. Vous devrez peut-être ajouter une stratégie réseau pour autoriser l'accès à l'IMDS EC2. Pour plus d'informations, consultez Métadonnées d'instance et données utilisateur dans le Guide de l'utilisateur Amazon EC2.

  Des pods qui utilisent des rôles IAM pour les comptes de service n'accèdent pas à l'IMDS EC2.
- Le Amazon VPC CNI plugin for Kubernetes n'applique pas les politiques réseau aux interfaces réseau supplémentaires pour chaque pod, mais uniquement à l'interface principale pour chaque pod (`eth0`). Cela concerne les architectures suivantes :
  - `IPv6` pods dont la variable `ENABLE_V4_EGRESS` est définie sur `true`. Cette variable permet à la fonctionnalité de sortie `IPv4` de connecter les pods IPv6 à des points de terminaison `IPv4`, comme ceux situés en dehors du cluster. La fonctionnalité de sortie `IPv4` fonctionne en créant une interface réseau supplémentaire avec une adresse IPv4 de boucle locale.

- Lorsque vous utilisez des plugins réseau enchaînés tels queMultus. Comme ces plug-ins ajoutent des interfaces réseau à chaque pod, les politiques réseau ne sont pas appliquées aux plug-ins réseau chaînés.

- La fonctionnalité de politique réseau utilise le port `8162` du nœud pour les métriques par défaut. En outre, la fonctionnalité utilisait le port `8163` pour les sondes d'intégrité. Si vous exécutez une autre application sur les nœuds ou dans les pods qui doivent utiliser ces ports, l'application ne s'exécutera pas. Dans la version VPC CNI `v1.14.1` ou une version ultérieure, vous pouvez modifier ces ports aux endroits suivants :

  AWS Management Console

  1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
  2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire Amazon VPC CNI.
  3. Choisissez l'onglet Modules complémentaires.
  4. Cochez la case en haut à droite de la zone du module complémentaire, puis sélectionnez Edit (Modifier).
  5. Sur la page Configurer ***nom du module complémentaire*** :

     a. Sélectionnez une version `v1.14.0-eksbuild.3` ou ultérieure dans la liste déroulante Version.

     b. Sélectionnez Paramètres de configuration facultatifs.

     c. Saisissez la clé JSON `"enableNetworkPolicy":` et valeur `"true"` dans Valeurs de configuration. Le texte obtenu doit être un objet JSON valide. Si cette clé et cette valeur sont les seules données de la zone de texte, entourez-les d'accolades `{}`.

        Dans l'exemple suivant, la fonctionnalité de politique réseau est activée, les journaux de politique réseau sont activés, les journaux de politique réseau sont envoyés à Amazon CloudWatch Logs, et les métriques et les sondes de santé sont définies sur les numéros de port par défaut :

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
```

```
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
        "healthProbeBindAddr": "8163",
        "metricsBindAddr": "8162"
    }
}
```

Helm

Si vous avez installé Amazon VPC CNI plugin for Kubernetes via `helm`, vous pouvez mettre à jour la configuration pour modifier les ports.

- Exécutez la commande suivante pour modifier les ports. Définissez le numéro de port dans la valeur de la clé `nodeAgent.metricsBindAddr` ou de la clé `nodeAgent.healthProbeBindAddr`.

```
helm upgrade --set nodeAgent.metricsBindAddr=8162 --set
 nodeAgent.healthProbeBindAddr=8163 aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Ouvrez le `aws-node DaemonSet` dans votre éditeur.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Remplacez les numéros de port dans les arguments de commande suivants pour le `args:` du conteneur `aws-network-policy-agent` dans le manifeste daemonset `aws-node` du VPC CNI.

```
    - args:
            - --metrics-bind-addr=:8162
            - --health-probe-bind-addr=:8163
```

Prérequis

- Version minimale du cluster

Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS. Le cluster doit être d'une version Kubernetes `1.25` ou ultérieure. Le cluster doit exécuter l'une des versions de Kubernetes et de la plateforme répertoriées dans le tableau suivant. Toute version de Kubernetes et de plateforme ultérieure à celles répertoriées est également prise en charge. Vous pouvez vérifier votre version Kubernetes actuelle en remplaçant *my-cluster* dans la commande suivante par le nom de votre cluster et en exécutant la commande :

```
aws eks describe-cluster
              --name my-cluster --query cluster.version --output
              text
```

Version de Kubernetes	Version de plateforme
1.27.4	eks.5
1.26.7	eks.6
1.25.12	eks.7

- Version minimale du VPC CNI

  Version `1.14` ou ultérieure du Amazon VPC CNI plugin for Kubernetes sur votre cluster. Vous pouvez voir la version que vous utilisez actuellement à l'aide de la commande suivante.

  ```
  kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
   cut -d : -f 3
  ```

  Si votre version est antérieure à `1.14`, consultez la section Mise à jour du module complémentaire Amazon EKS pour mettre à jour vers la version `1.14` ou ultérieure.

- Version minimale du noyau Linux

  Vos nœuds doivent avoir la version du noyau Linux `5.10` ou ultérieure. Vous pouvez vérifier votre version du noyau avec `uname -r`. Si vous utilisez les dernières versions d'Amazon Linux optimisé pour Amazon EKS, des AMI Amazon Linux accéléré optimisé pour Amazon EKS et des AMI Bottlerocket, elles disposent déjà de la version du noyau requise.

  La version `v20231116` ou ultérieure de l'AMI Amazon Linux accélérée et optimisée pour Amazon EKS dispose de la version `5.10` du noyau.

Pour configurer votre cluster afin d'utiliser les stratégies réseau Kubernetes

1. Monter le système de fichiers BPF

> **Note**
>
> Si votre cluster est une version `1.27` ou ultérieure, vous pouvez ignorer cette étape car toutes les AMI Amazon Linux et Bottlerocket optimisées par Amazon EKS pour la version `1.27` ou ultérieure, vous avez déjà cette fonctionnalité.
> Pour toutes les autres versions de cluster, si vous mettez à niveau Amazon Linux optimisé pour Amazon EKS vers la version `v20230703` ou ultérieure ou vous mettez à niveau l'AMI Bottlerocket vers la version `v1.0.2` ou ultérieure, vous pouvez ignorer cette étape.

   a. Montez le système de fichiers Berkeley Packet Filter (BPF) sur chacun de vos nœuds.

```
sudo mount -t bpf bpffs /sys/fs/bpf
```

   b. Ajoutez ensuite la même commande à vos données utilisateur dans votre modèle de lancement pour vos groupes Amazon EC2 Auto Scaling.

2. Activer la stratégie réseau dans le VPC CNI

   a. Déterminez le type de module complémentaire installé sur votre cluster. Selon l'outil avec lequel vous avez créé votre cluster, le type de module complémentaire Amazon EKS peut ne pas être actuellement installé sur votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

   Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster, et vous n'avez donc pas besoin de suivre les étapes restantes de cette procédure. Si une erreur est renvoyée, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster.

   b. • Module complémentaire d'Amazon EKS

AWS Management Console

a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

b. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire Amazon VPC CNI.

c. Choisissez l'onglet Modules complémentaires.

d. Cochez la case en haut à droite de la zone du module complémentaire, puis sélectionnez Edit (Modifier).

e. Sur la page Configurer ***nom du module complémentaire*** :

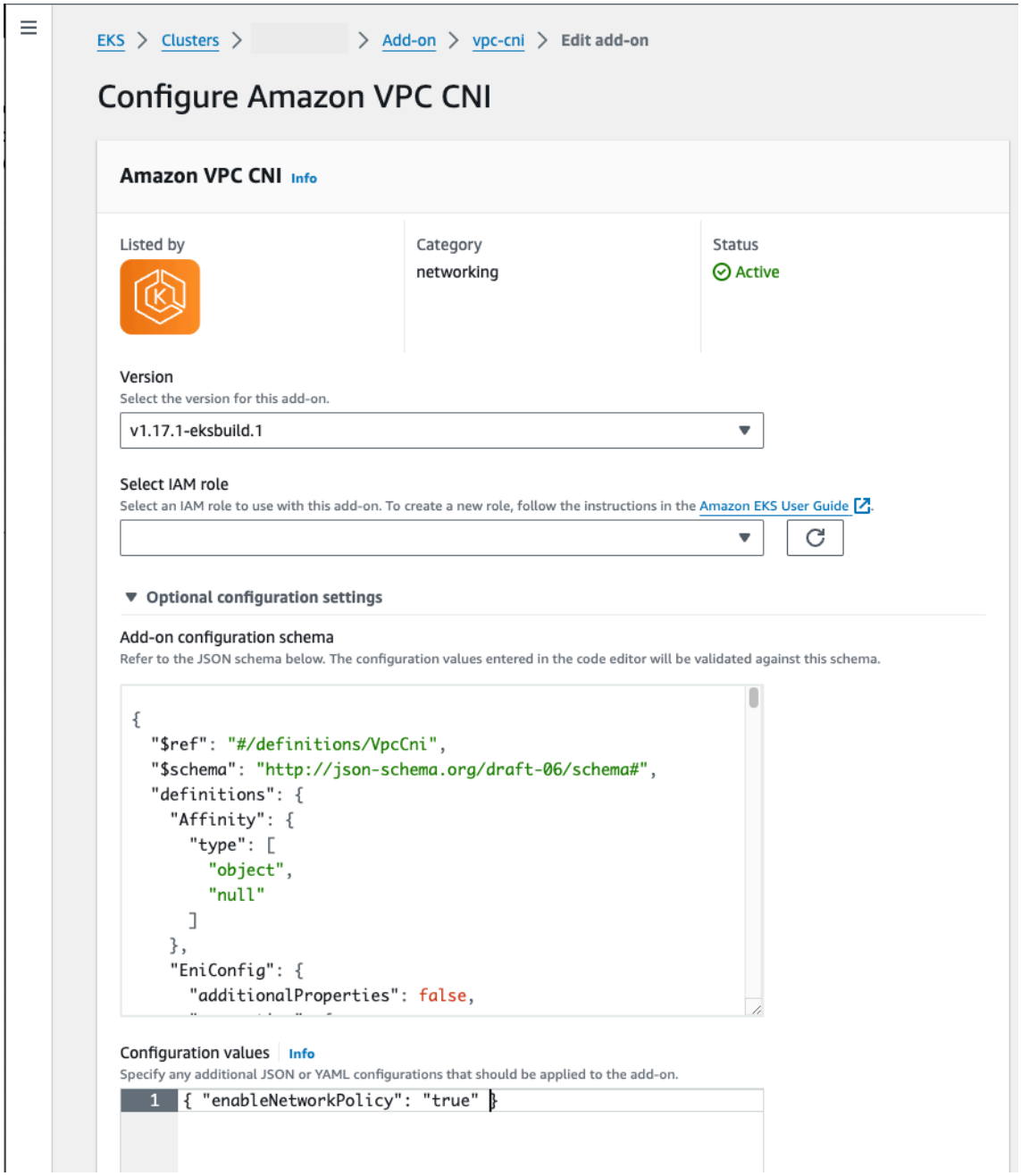
   i. Sélectionnez une version `v1.14.0-eksbuild.3` ou ultérieure dans la liste déroulante Version.

   ii. Sélectionnez Paramètres de configuration facultatifs.

   iii. Saisissez la clé JSON `"enableNetworkPolicy":` et valeur `"true"` dans Valeurs de configuration. Le texte obtenu doit être un objet JSON valide. Si cette clé et cette valeur sont les seules données de la zone de texte, entourez-les d'accolades `{}`. L'exemple suivant montre que la politique réseau est activée :

   ```
   { "enableNetworkPolicy": "true" }
   ```

   La capture d'écran suivante montre un exemple de ce scénario.



AWS CLI

- Exécutez la AWS CLI commande suivante. Remplacez `my-cluster` par le nom de votre cluster et l'ARN du rôle IAM par le rôle que vous utilisez.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni
 --addon-version v1.14.0-eksbuild.3 \
```

```
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values
 '{"enableNetworkPolicy": "true"}'
```

- Module complémentaire autogéré

  Helm

  Si vous avez installé le Amazon VPC CNI plugin for Kubernetes à travers `helm`, vous pouvez mettre à jour la configuration pour activer la stratégie réseau.

  - Exécutez la commande suivante pour activer la stratégie réseau.

    ```
    helm upgrade --set enableNetworkPolicy=true aws-vpc-cni --namespace
     kube-system eks/aws-vpc-cni
    ```

  kubectl

  a. Ouvrez le `amazon-vpc-cni` `ConfigMap` dans votre éditeur.

     ```
     kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
     ```

  b. Ajoutez la ligne suivante dans les `data` de `ConfigMap`.

     ```
     enable-network-policy-controller: "true"
     ```

     Une fois que vous avez ajouté la ligne, votre `ConfigMap` devrait ressembler à l'exemple suivant.

     ```
     apiVersion: v1
      kind: ConfigMap
      metadata:
       name: amazon-vpc-cni
       namespace: kube-system
      data:
       enable-network-policy-controller: "true"
     ```

  c. Ouvrez le `aws-node` `DaemonSet` dans votre éditeur.

```
kubectl edit daemonset -n kube-system aws-node
```

d. Remplacez `false` par `true` dans l'argument de commande `--enable-network-policy=false` pour le `args:` du conteneur `aws-network-policy-agent` dans le manifeste daemonset du `aws-node` VPC CNI.

```
    - args:
       - --enable-network-policy=true
```

3. Confirmez que les pods du `aws-node` sont en cours d'exécution sur votre cluster.

```
kubectl get pods -n kube-system | grep 'aws-node\|amazon'
```

L'exemple qui suit illustre un résultat.

```
aws-node-gmqp7                                          2/2     Running   1 (24h
 ago)   24h
aws-node-prnsh                                          2/2     Running   1 (24h
 ago)   24h
```

Si la stratégie réseau est activée, il existe 2 conteneurs dans les pods du `aws-node`. Dans les versions précédentes et si la stratégie réseau est désactivée, il n'y a qu'un seul conteneur dans les pods du `aws-node`.

Vous pouvez désormais déployer des stratégies réseau Kubernetes à votre cluster. Pour plus d'informations, consultez Stratégies réseau Kubernetes.

Démonstration de la stratégie réseau Stars

Cette démonstration permet de créer un service client, un service frontal et un service backend sur votre cluster Amazon EKS. La démonstration crée également une interface utilisateur graphique de gestion qui affiche les chemins d'entrée et de sortie disponibles entre chaque service. Nous vous recommandons de terminer la démonstration sur un cluster sur lequel vous n'exécutez pas de charges de travail de production.

Avant de créer des politiques réseau, tous les services peuvent communiquer de façon bidirectionnelle. Une fois que vous avez appliqué les politiques réseau, vous constatez que le client

peut uniquement communiquer avec le service frontend et que le service backend n'accepte que le trafic du frontend.

Pour exécuter la démonstration de politique Stars

1. Appliquez les services frontal, backend et d'interface utilisateur de gestion :

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
```

2. Affichez tous les Pods du cluster.

```
kubectl get pods -A
```

L'exemple qui suit illustre un résultat.

Dans votre sortie, vous devriez voir des pods dans les espaces de noms affichés dans la sortie suivante. Les *NOMS* de vos pods et le nombre de pods dans la colonne READY sont différents de ceux de la sortie suivante. Ne continuez pas jusqu'à ce que vous voyiez des pods avec des noms similaires et qu'ils aient tous Running dans la colonne STATUS.

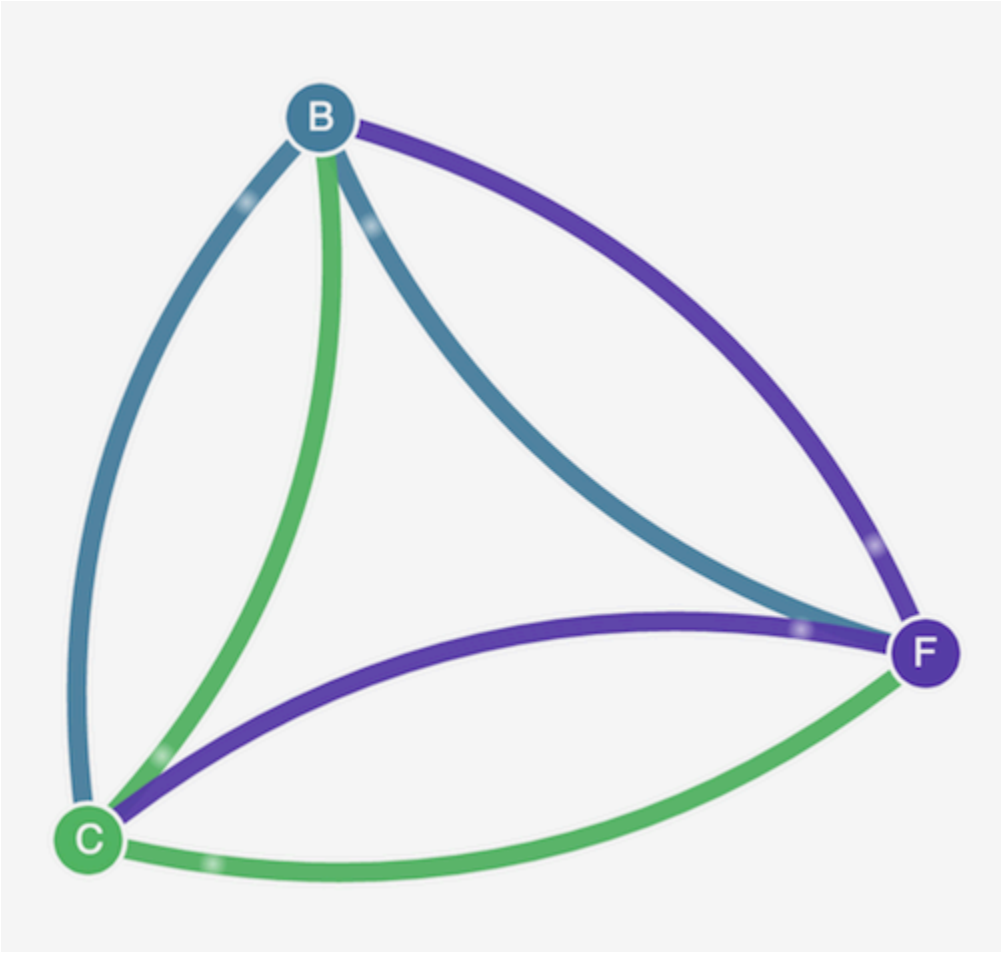
```
NAMESPACE         NAME                                       READY   STATUS
 RESTARTS   AGE
[...]
client            client-xlffc                               1/1     Running   0
       5m19s
[...]
management-ui     management-ui-qrb2g                        1/1     Running   0
       5m24s
stars             backend-sz87q                              1/1     Running   0
       5m23s
stars             frontend-cscnf                             1/1     Running   0
       5m21s
```

```
[...]
```

3. Pour vous connecter à l'interface utilisateur de gestion, connectez-vous à l'`EXTERNAL-IP` du service qui s'exécute sur votre cluster :

```
kubectl get service/management-ui -n management-ui
```

4. Ouvrez le navigateur à l'emplacement indiqué à l'étape précédente. Vous devez voir l'interface utilisateur de gestion. Le nœud C est le service client, le nœud F est le service frontal et le nœud B est le service backend. Chaque nœud a un accès en communication complet à tous les autres nœuds, comme indiqué par les lignes colorées en gras.



5. Appliquez les stratégies réseau suivantes tant dans les espaces de noms `stars` que `client` pour isoler les services les uns des autres :

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny
```

```
spec:
  podSelector:
    matchLabels: {}
```

Vous pouvez utiliser les commandes suivantes pour appliquer la stratégie aux deux espaces de noms :

```
kubectl apply -n stars -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
kubectl apply -n client -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
```

6. Actualisez votre navigateur. Vous pouvez alors constater que l'interface utilisateur graphique de gestion ne peut plus atteindre aucun des nœuds. C'est pour cette raison qu'ils n'apparaissent pas dans l'interface utilisateur graphique.
7. Appliquez les stratégies réseau différentes suivantes pour permettre à l'interface utilisateur de gestion d'accéder aux services. Appliquez cette stratégie pour autoriser l'interface utilisateur :

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Appliquez cette stratégie pour autoriser le client :

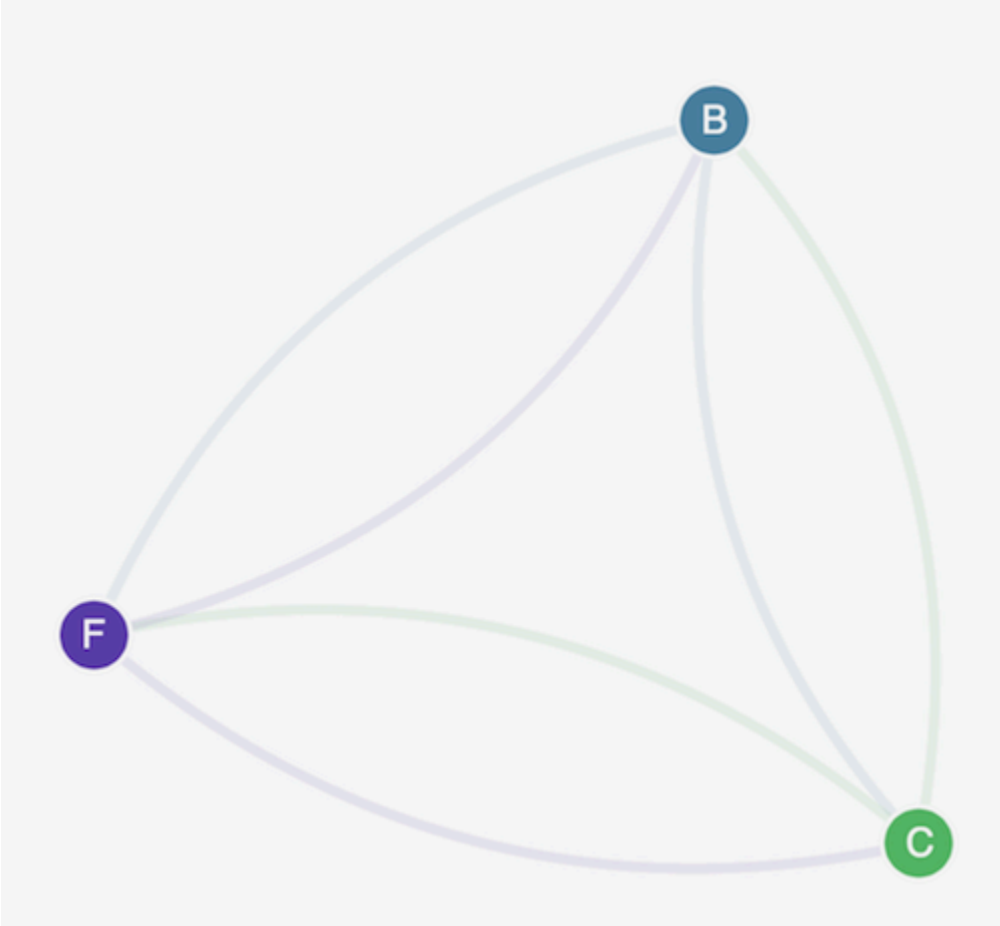
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
```

```
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Vous pouvez utiliser les commandes suivantes pour appliquer les deux stratégies :

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui-client.yaml
```

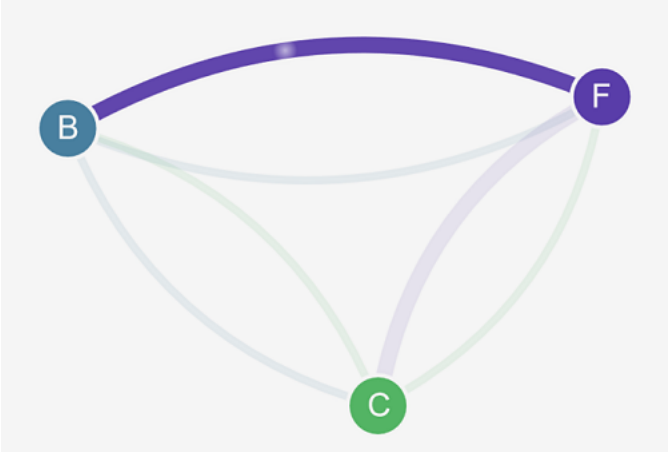
8. Actualisez votre navigateur. Vous pouvez alors constater que l'interface utilisateur graphique de gestion peut à nouveau atteindre les nœuds, mais que les nœuds ne peuvent pas communiquer entre eux.



9. Appliquez la politique réseau suivante pour autoriser le trafic du service frontal au service backend :

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
```

10. Actualisez votre navigateur. Vous voyez que le frontend peut communiquer avec le backend.

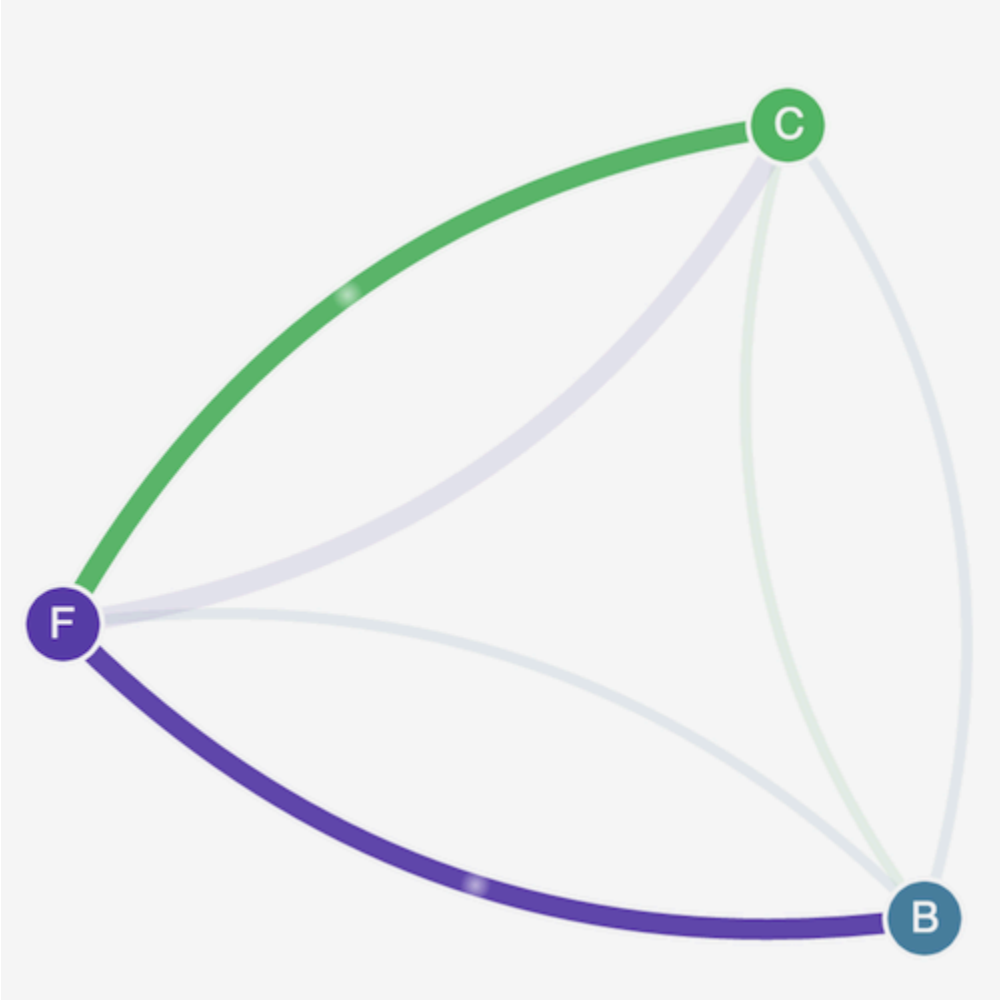


11. Appliquez la stratégie réseau suivante pour autoriser le trafic du client vers le service frontend :

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
```

```
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: TCP
          port: 80
```

12. Actualisez votre navigateur. Vous voyez que le client peut communiquer avec le service frontend. Le service frontend peut toujours communiquer avec le service backend.



13. (Facultatif) Une fois que vous avez terminé avec la démonstration, vous pouvez supprimer ses ressources.

```
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
```

Même après la suppression des ressources, il peut toujours y avoir des points de terminaison de stratégie réseau sur les nœuds qui peuvent interférer de manière inattendue avec la mise en réseau dans votre cluster. La seule façon sûre de supprimer ces règles est de redémarrer les nœuds ou de terminer tous les nœuds et de les recycler. Pour mettre fin à tous les nœuds, définissez le nombre souhaité du groupe Auto Scaling sur 0, puis sauvegardez jusqu'au nombre souhaité, ou terminez simplement les nœuds.

Dépannage des stratégies réseau

Vous pouvez résoudre les problèmes et étudier les connexions réseau qui utilisent des stratégies réseau en lisant le Journaux de stratégie réseau et en exécutant les outils du SDK eBPF.

Journaux de stratégie réseau

Si les connexions sont autorisées ou refusées par un réseau, les stratégies sont enregistrées dans les journaux de flux. Les journaux de stratégie réseau de chaque nœud incluent les journaux de flux pour chaque pod doté d'une stratégie réseau. Les journaux de stratégie réseau sont stockés sur `/var/log/aws-routed-eni/network-policy-agent.log`. L'exemple suivant est extrait d'un fichier `network-policy-agent.log` :

```
{"level":"info","timestamp":"2023-05-30T16:05:32.573Z","logger":"ebpf-
client","msg":"Flow Info: ","Src
IP":"192.168.87.155","Src Port":38971,"Dest IP":"64.6.160","Dest
Port":53,"Proto":"UDP","Verdict":"ACCEPT"}
```

Les journaux de politique réseau sont désactivés par défaut. Pour activer les journaux de politique réseau, procédez comme suit :

> **Note**
>
> Les journaux de politique réseau nécessitent 1 vCPU supplémentaire pour le `aws-network-policy-agent` conteneur dans le manifeste du daemonset VPC CNI. `aws-node`

Module complémentaire d'Amazon EKS

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire Amazon VPC CNI.
3. Choisissez l'onglet Modules complémentaires.
4. Cochez la case en haut à droite de la zone du module complémentaire, puis sélectionnez Edit (Modifier).
5. Sur la page Configurer ***nom du module complémentaire*** :

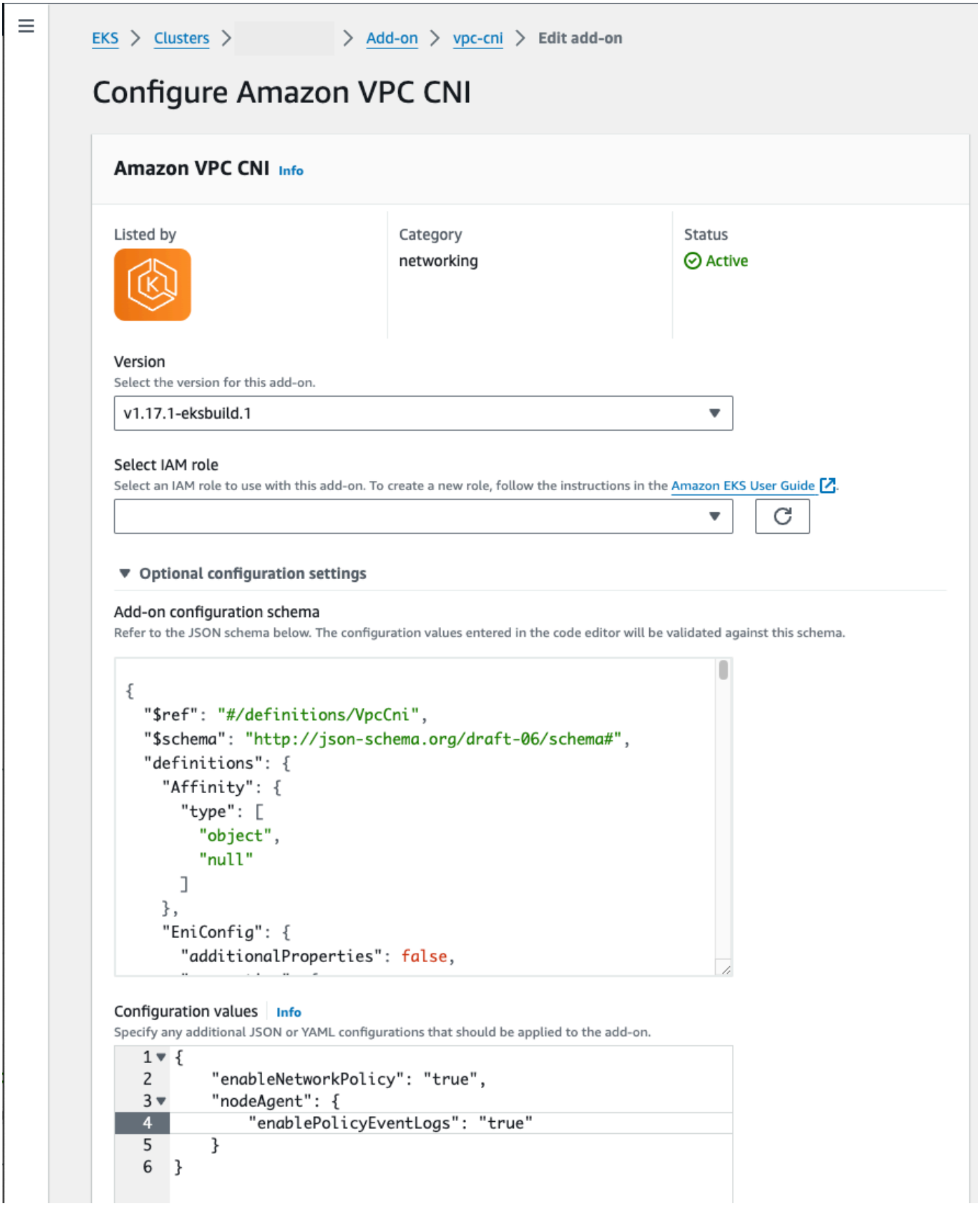
   a. Sélectionnez une version `v1.14.0-eksbuild.3` ou ultérieure dans la liste déroulante Version.

   b. Sélectionnez Paramètres de configuration facultatifs.

   c. Saisissez la clé JSON de niveau supérieur de l'`"nodeAgent":` et la valeur est un objet avec une clé `"enablePolicyEventLogs":` et une valeur de `"true"` dansValeurs de configuration. Le texte obtenu doit être un objet JSON valide. L'exemple suivant montre la politique réseau et les journaux de stratégie réseau sont activés, et les journaux de stratégie réseau sont envoyés à CloudWatch Logs :

   ```
   {
       "enableNetworkPolicy": "true",
       "nodeAgent": {
           "enablePolicyEventLogs": "true"
       }
   }
   ```

La capture d'écran suivante montre un exemple de ce scénario.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true"
5 }
6 }

AWS CLI

- Exécutez la AWS CLI commande suivante. Remplacez `my-cluster` par le nom de votre cluster et l'ARN du rôle IAM par le rôle que vous utilisez.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true"}}'
```

Module complémentaire autogéré

Helm

Si vous avez installé le Amazon VPC CNI plugin for Kubernetes through`helm`, vous pouvez mettre à jour la configuration pour écrire les journaux de politique réseau.

- Exécutez la commande suivante pour activer la stratégie réseau.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true aws-vpc-cni --namespace
 kube-system eks/aws-vpc-cni
```

kubectl

Si vous avez installé le Amazon VPC CNI plugin for Kubernetes through`kubectl`, vous pouvez mettre à jour la configuration pour écrire les journaux de politique réseau.

1. Ouvrez le `aws-node` `DaemonSet` dans votre éditeur.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Remplacez `false` par `true` dans l'argument de commande `--enable-policy-event-logs=false` pour le `args:` du conteneur `aws-network-policy-agent` dans le manifeste daemonset du `aws-node` VPC CNI.

```
     - args:
```

```
        - --enable-policy-event-logs=true
```

Envoyer des journaux de politique réseau à Amazon CloudWatch Logs

Vous pouvez surveiller les journaux de politique réseau à l'aide de services tels qu'Amazon CloudWatch Logs. Vous pouvez utiliser les méthodes suivantes pour envoyer les journaux de politique réseau à CloudWatch Logs.

Pour les clusters EKS, les journaux de stratégie seront situés sous `/aws/eks/`*`cluster-name`*`/cluster/` et pour les clusters K8S autogérés, les journaux seront placés sous `/aws/k8s-cluster/cluster/`.

Envoyez les journaux de stratégie réseau avec le Amazon VPC CNI plugin for Kubernetes

Si vous activez la stratégie réseau, un deuxième conteneur est ajouté aux pods du `aws-node` pour unagent de nœud. Cet agent de nœud peut envoyer les journaux de politique réseau à CloudWatch Logs.

> **Note**
>
> Seuls les journaux de stratégie réseau sont envoyés par l'agent de nœud. Les autres journaux créés par le VPC CNI ne sont pas inclus.

Prérequis

- Ajoutez les autorisations suivantes sous forme de strophe ou de stratégie distincte au rôle IAM que vous utilisez pour le VPC CNI.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Module complémentaire d'Amazon EKS

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire Amazon VPC CNI.
3. Choisissez l'onglet Modules complémentaires.
4. Cochez la case en haut à droite de la zone du module complémentaire, puis sélectionnez Edit (Modifier).
5. Sur la page Configurer ***nom du module complémentaire*** :

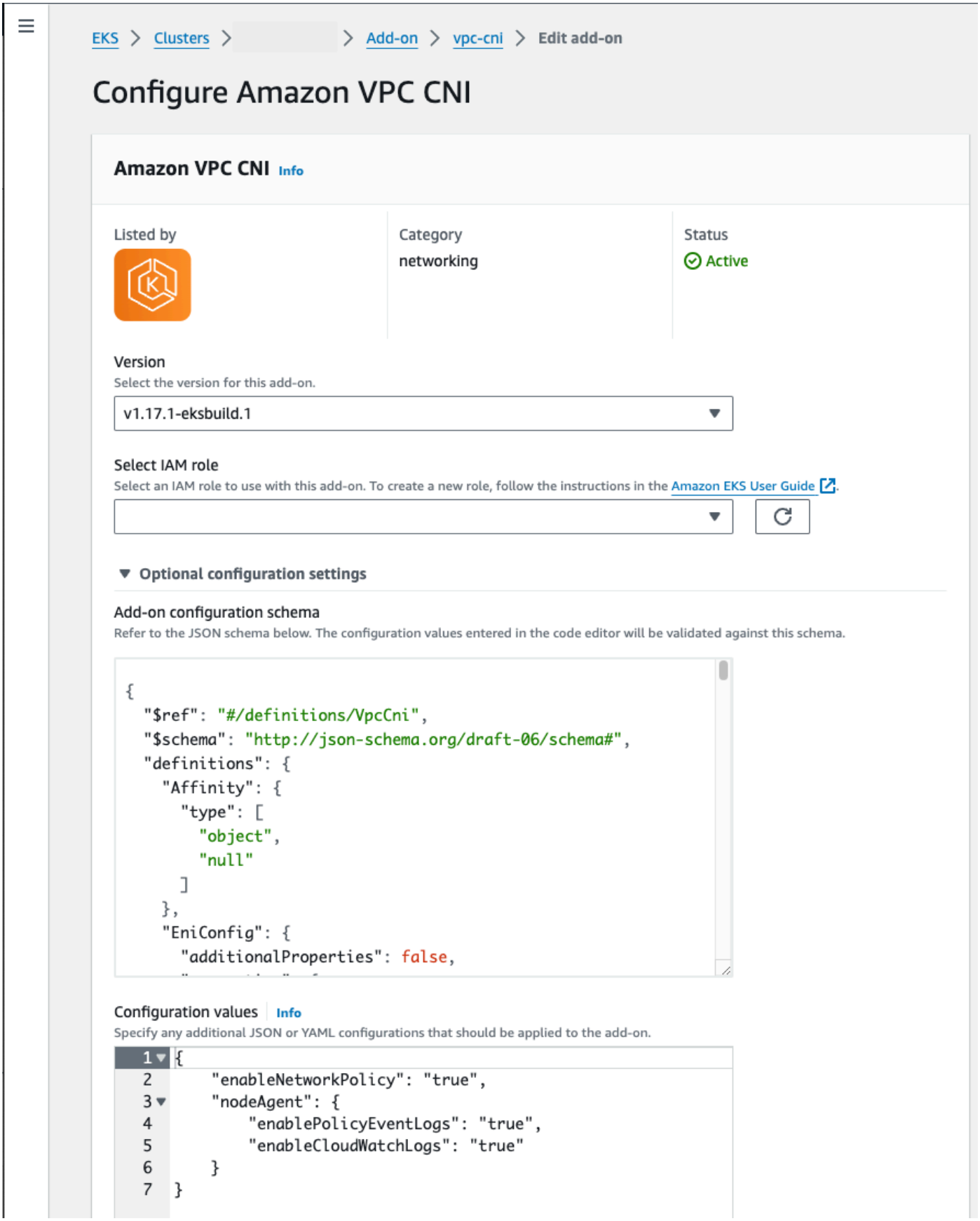
    a. Sélectionnez une version `v1.14.0-eksbuild.3` ou ultérieure dans la liste déroulante Version.

    b. Sélectionnez Paramètres de configuration facultatifs.

    c. Saisissez la clé JSON de niveau supérieur de l'`"nodeAgent":` et la valeur est un objet avec une clé `"enableCloudWatchLogs":` et une valeur de `"true"` dansValeurs de configuration. Le texte obtenu doit être un objet JSON valide. L'exemple suivant montre la politique réseau, les journaux de stratégie réseau sont activés et les journaux sont envoyés à CloudWatch Logs :

    ```
    {
        "enableNetworkPolicy": "true",
        "nodeAgent": {
            "enablePolicyEventLogs": "true",
            "enableCloudWatchLogs": "true",
        }
    }
    ```

La capture d'écran suivante montre un exemple de ce scénario.


EKS > Clusters > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true",
5 "enableCloudWatchLogs": "true"
6 }
7 }

AWS CLI

- Exécutez la AWS CLI commande suivante. Remplacez `my-cluster` par le nom de votre cluster et l'ARN du rôle IAM par le rôle que vous utilisez.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true", "enableCloudWatchLogs": "true"}}'
```

Module complémentaire autogéré

Helm

Si vous avez installé le Amazon VPC CNI plugin for Kubernetes throughhelm, vous pouvez mettre à jour la configuration pour envoyer les journaux de politique réseau à CloudWatch Logs.

- Exécutez la commande suivante pour activer les journaux de politique réseau et les envoyer à CloudWatch Logs.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true --set
 nodeAgent.enableCloudWatchLogs=true aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Ouvrez le `aws-node` `DaemonSet` dans votre éditeur.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Remplacez le `false` par `true` dans deux arguments de commande `--enable-policy-event-logs=false` et `--enable-cloudwatch-logs=false` dans le `aws-network-policy-agent` conteneur `args:` dans le manifeste du daemonset VPC CNI`aws-node`.

```
    - args:
       - --enable-policy-event-logs=true
```

```
        - --enable-cloudwatch-logs=true
```

Envoyez les journaux de stratégie réseau avec un daemonset Fluent Bit

Si vous utilisez Fluent Bit dans un daemonset pour envoyer des journaux à partir de vos nœuds, vous pouvez ajouter une configuration pour inclure les journaux de politique réseau à partir des politiques réseau. Vous pouvez utiliser l'exemple de configuration suivant :

```
    [INPUT]
        Name              tail
        Tag               eksnp.*
        Path              /var/log/aws-routed-eni/network-policy-agent*.log
        Parser            json
        DB                /var/log/aws-routed-eni/flb_npagent.db
        Mem_Buf_Limit     5MB
        Skip_Long_Lines   On
        Refresh_Interval  10
```

Inclus SDK eBPF

Le Amazon VPC CNI plugin for Kubernetes installe l'ensemble d'outils du SDK eBPF sur les nœuds. Vous pouvez utiliser les outils du SDK eBPF pour identifier les problèmes liés aux stratégies réseau. Par exemple, la commande suivante répertorie les programmes qui s'exécutent sur le nœud.

```
sudo /opt/cni/bin/aws-eks-na-cli ebpf progs
```

Pour exécuter cette commande, vous pouvez utiliser n'importe quelle méthode pour vous connecter au nœud.

Stratégies réseau Kubernetes

Pour mettre en œuvre des stratégies réseau Kubernetes, vous créez des objets Kubernetes `NetworkPolicy` et les déployez sur votre cluster. Les objets `NetworkPolicy` sont limités à un espace de noms. Vous mettez en œuvre des stratégies pour autoriser ou refuser le trafic entre les Pods en fonction des sélecteurs d'étiquettes, des espaces de noms et des plages d'adresses IP. Pour plus d'informations sur la création d'objets `NetworkPolicy`, consultez Stratégies réseau dans la documentation Kubernetes.

L'application des objets `NetworkPolicy` Kubernetes est mise en œuvre à l'aide du Extended Berkeley Packet Filter (eBPF). Concernant les implémentations basées sur les `iptables`, il offre

des caractéristiques de latence et de performance plus faibles, notamment une utilisation réduite du processeur et l'évitement des recherches séquentielles. De plus, les sondes eBPF fournissent un accès à des données contextuelles qui aident à résoudre les problèmes complexes au niveau du noyau et à améliorer l'observabilité. Amazon EKS prend en charge un exportateur basé sur eBPF qui exploite les sondes pour enregistrer les résultats des stratégies sur chaque nœud et exporter les données vers des collecteurs de journaux externes afin de faciliter le débogage. Pour en savoir plus, consultez la documentation eBPF.

Mise en réseau personnalisée pour les pods

Par défaut, lorsque le Amazon VPC CNI plugin for Kubernetes crée des interfaces réseau élastiques secondaires (interfaces réseau) pour votre nœud Amazon EC2, il les crée dans le même sous-réseau que l'interface réseau principale du nœud. Il associe également les mêmes groupes de sécurité à l'interface réseau secondaire, qui sont associés à l'interface réseau principale. Pour une ou plusieurs des raisons suivantes, vous voudrez peut-être que le plugin crée des interfaces réseau secondaires dans un sous-réseau différent ou associe différents groupes de sécurité aux interfaces réseau secondaires, ou les deux :

- Il y a un nombre limité d'adresses `IPv4` disponibles dans le sous-réseau dans lequel se trouve l'interface réseau principale. Cela peut limiter le nombre de Pods que vous pouvez créer dans le sous-réseau. En utilisant un sous-réseau différent pour les interfaces réseau secondaires, vous pouvez augmenter le nombre d'adresses `IPv4` disponibles pour les Pods.
- Pour des raisons de sécurité, vos Pods doivent utiliser d'autres groupes de sécurité ou sous-réseaux que l'interface réseau principale du nœud.
- Les nœuds sont configurés dans des sous-réseaux publics et vous souhaitez placer les Pods dans des sous-réseaux privés. La table de routage associée à un sous-réseau public comprend une route vers une passerelle Internet. La table de routage associée à un sous-réseau privé ne comprend pas une route vers une passerelle Internet.

Considérations

- Lorsque la mise en réseau personnalisée est activée, aucune adresse IP attribuée à l'interface réseau principale n'est affectée aux Pods. Seules les adresses IP des interfaces réseau secondaires sont attribuées aux `Pods`.
- Si votre cluster utilise la famille `IPv6`, vous ne pouvez pas utiliser de mise en réseau personnalisée.

- Si vous prévoyez d'utiliser une mise en réseau personnalisée uniquement pour pallier à l'épuisement des adresses `IPv4`, vous pouvez plutôt créer un cluster à l'aide de la famille `IPv6`. Pour plus d'informations, consultez `IPv6`adresses pour les clustersPods, et services.
- Même si les Pods déployés sur des sous-réseaux spécifiés pour les interfaces réseau secondaires peuvent utiliser des sous-réseaux et des groupes de sécurité différents de ceux de l'interface réseau principale du nœud, les sous-réseaux et les groupes de sécurité doivent se trouver dans le même VPC que le nœud.

Prérequis

- Maîtriser la façon dont le Amazon VPC CNI plugin for Kubernetes crée des interfaces réseau secondaires et attribue des adresses IP aux Pods. Pour de plus amples informations, veuillez consulter Allocation ENI sur GitHub.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Nous vous recommandons de terminer les étapes de cette rubrique dans un shell Bash. Si vous n'utilisez pas de shell Bash, certaines commandes de script telles que les caractères de continuation de ligne et la façon dont les variables sont définies et utilisées nécessitent un ajustement pour votre shell. En outre, les règles de votre shell en matière de guillemets peuvent être différentes. Pour plus d'informations, consultez la section Utilisation de guillemets avec des chaînes AWS CLI dans le Guide de AWS Command Line Interface l'utilisateur.

Pour ce tutoriel, nous vous recommandons d'utiliser les *example values*, sauf lorsqu'il est noté de les remplacer. Vous pouvez remplacer n'importe quel *example value* lors des étapes propres à un cluster de production. Nous vous recommandons de terminer toutes les étapes dans le même terminal. En effet, les variables sont définies et utilisées tout au long des étapes et n'existent pas dans différents terminaux.

Les commandes de cette rubrique sont mises en forme selon les conventions répertoriées dans Utilisation des AWS CLI exemples. Si vous exécutez des commandes depuis la ligne de commande sur des ressources dont la valeur est différente de Région AWS celle Région AWS définie par défaut dans le AWS CLI profil que vous utilisez, vous devez les ajouter **--region** *region-code* aux commandes.

Lorsque vous souhaitez déployer une mise en réseau personnalisée sur votre cluster de production, passez à Étape 2 : configuration de votre VPC.

Étape 1 : créer un VPC test et un cluster

Pour créer un cluster

Les procédures suivantes vous aident à créer un VPC test et un cluster et à configurer une mise en réseau personnalisée pour ce cluster. Nous ne recommandons pas d'utiliser le cluster test pour les charges de travail de production, car plusieurs fonctionnalités non liées que vous pouvez utiliser sur votre cluster de production ne sont pas couvertes dans cette rubrique. Pour plus d'informations, consultez Création d'un cluster Amazon EKS.

1. Définissez quelques variables à utiliser dans les étapes restantes.

```
export cluster_name=my-custom-networking-cluster
account_id=$(aws sts get-caller-identity --query Account --output text)
```

2. Créez un VPC.

   1. Créez un VPC à l'aide d'un modèle Amazon EKS AWS CloudFormation .

```
aws cloudformation create-stack --stack-name my-eks-custom-networking-vpc \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --parameters ParameterKey=VpcBlock,ParameterValue=192.168.0.0/24 \
  ParameterKey=PrivateSubnet01Block,ParameterValue=192.168.0.64/27 \
  ParameterKey=PrivateSubnet02Block,ParameterValue=192.168.0.96/27 \
  ParameterKey=PublicSubnet01Block,ParameterValue=192.168.0.0/27 \
```

```
  ParameterKey=PublicSubnet02Block,ParameterValue=192.168.0.32/27
```

La création de la AWS CloudFormation pile prend quelques minutes. Utilisez la commande suivante pour vérifier le statut du déploiement de la pile.

```
aws cloudformation describe-stacks --stack-name my-eks-custom-networking-vpc --
query Stacks\[\].StackStatus  --output text
```

Ne passez pas à l'étape suivante tant que la sortie de la commande n'est pas `CREATE_COMPLETE`.

2. Définissez des variables avec les valeurs des ID de sous-réseau privé créés par le modèle.

```
subnet_id_1=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet01'].PhysicalResourceId" --output text)
subnet_id_2=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet02'].PhysicalResourceId" --output text)
```

3. Définissez des variables avec les zones de disponibilité des sous-réseaux récupérés lors de l'étape précédente.

```
az_1=$(aws ec2 describe-subnets --subnet-ids $subnet_id_1 --query
 'Subnets[*].AvailabilityZone' --output text)
az_2=$(aws ec2 describe-subnets --subnet-ids $subnet_id_2 --query
 'Subnets[*].AvailabilityZone' --output text)
```

3. Créez un rôle IAM de cluster.

   a. Exécutez la commande suivante pour créer un fichier JSON de politique d'approbation IAM.

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

b. Créez le rôle IAM de cluster Amazon EKS. Si nécessaire, faites précéder `eks-cluster-role-trust-policy.json` par le chemin d'accès sur votre ordinateur sur lequel vous avez enregistré le fichier lors de l'étape précédente. La commande associe la politique d'approbation que vous avez créée lors de l'étape précédente à ce rôle. Pour créer un rôle IAM, le principal IAM qui crée le rôle doit se voir attribuer l'action (autorisation) IAM `iam:CreateRole`.

```
aws iam create-role --role-name myCustomNetworkingAmazonEKSClusterRole --
assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

c. Attachez la politique gérée par Amazon EKS appelée `AmazonEKSClusterPolicy` au rôle. Pour attacher une politique IAM à un principal IAM, le principal qui attache la politique doit se voir attribuer l'une des actions (autorisations) IAM `iam:AttachUserPolicy` ou `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name myCustomNetworkingAmazonEKSClusterRole
```

4. Créez un cluster Amazon EKS et configurez votre appareil pour qu'il communique avec lui.

a. Créer un cluster.

```
aws eks create-cluster --name my-custom-networking-cluster \
   --role-arn arn:aws:iam::$account_id:role/
myCustomNetworkingAmazonEKSClusterRole \
   --resources-vpc-config subnetIds=$subnet_id_1","$subnet_id_2
```

> **Note**
>
> Il est possible que vous receviez un message d'erreur indiquant que l'une des zones de disponibilité de votre demande ne dispose pas d'une capacité suffisante pour créer un cluster Amazon EKS. Si cela se produit, la sortie de l'erreur contient les

zones de disponibilité qui peuvent prendre en charge un nouveau cluster. Essayez à nouveau de créer votre cluster avec au moins deux sous-réseaux situés dans les zones de disponibilité prises en charge pour votre compte. Pour plus d'informations, consultez Capacité insuffisante.

b. La création du cluster prend quelques minutes. Utilisez la commande suivante pour vérifier le statut du déploiement du cluster.

```
aws eks describe-cluster --name my-custom-networking-cluster --query
 cluster.status
```

Ne passez pas à l'étape suivante tant que la sortie de la commande n'est pas `"ACTIVE"`.

c. Configurez `kubectl` pour communiquer avec votre cluster.

```
aws eks update-kubeconfig --name my-custom-networking-cluster
```

Étape 2 : configuration de votre VPC

Dans ce didacticiel, vous devez disposer du VPC créé dans Étape 1 : créer un VPC test et un cluster. Pour un cluster de production, ajustez les étapes en conséquence pour votre VPC en remplaçant toutes les *`example values`* par vos propres valeurs.

1. Confirmez que la version de votre Amazon VPC CNI plugin for Kubernetes actuellement installée est la plus récente. Pour déterminer la version la plus récente du type de module complémentaire Amazon EKS et mettre à jour votre version en conséquence, consultez Mise à jour d'un module complémentaire. Pour déterminer la version la plus récente du type de module complémentaire autogéré et mettre à jour votre version en conséquence, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.
2. Récupérez l'ID du VPC de votre cluster et stockez-le dans une variable pour une utilisation ultérieure. Pour un cluster de production, remplacez *`my-custom-networking-cluster`* par le nom de votre cluster.

```
vpc_id=$(aws eks describe-cluster --name my-custom-networking-cluster --query
 "cluster.resourcesVpcConfig.vpcId" --output text)
```

3. Associez un bloc de routage inter-domaines sans classe (CIDR) supplémentaire au VPC de votre cluster. Le bloc CIDR ne peut pas se chevaucher avec des blocs CIDR associés existants.

1. Affichez les blocs CIDR actuels associés à votre VPC.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id \
    --query 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

   L'exemple qui suit illustre un résultat.

```
---------------------------------
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
|  192.168.0.0/24 |  associated  |
+-----------------+--------------+
```

2. Associez un bloc CIDR supplémentaire à votre VPC. Pour plus d'informations, veuillez consulter la section Associer des blocs d'adresse CIDR IPv4 supplémentaires à votre VPC dans le Guide de l'utilisateur Amazon VPC.

```
aws ec2 associate-vpc-cidr-block --vpc-id $vpc_id --cidr-block 192.168.1.0/24
```

3. Vérifiez que le nouveau bloc est associé.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id --query
 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

   L'exemple qui suit illustre un résultat.

```
---------------------------------
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
|  192.168.0.0/24 |  associated  |
|  192.168.1.0/24 |  associated  |
+-----------------+--------------+
```

Ne passez pas à l'étape suivante tant que le `State` de votre nouveau bloc CIDR n'est pas `associated`.

4. Créez autant de sous-réseaux que vous souhaitez utiliser dans chaque zone de disponibilité dans laquelle se trouvent vos sous-réseaux existants. Spécifiez un bloc CIDR qui se trouve dans le bloc CIDR que vous avez associé à votre VPC lors d'une étape précédente.

   1. Créez de nouveaux sous-réseaux. Les sous-réseaux doivent être créés dans un bloc CIDR VPC différent de celui de vos sous-réseaux existants, mais dans les mêmes zones de disponibilité que vos sous-réseaux existants. Dans cet exemple, un sous-réseau est créé dans le nouveau bloc CIDR de chaque zone de disponibilité dans laquelle les sous-réseaux privés actuels existent. Les ID des sous-réseaux créés sont stockés dans des variables pour être utilisés ultérieurement. Les valeurs `Name` correspondent aux valeurs affectées aux sous-réseaux créés à l'aide du modèle Amazon EKS VPC lors d'une étape précédente. Les noms ne sont pas obligatoires. Vous pouvez utiliser des noms différents.

```
new_subnet_id_1=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_1 --cidr-block 192.168.1.0/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet01},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
new_subnet_id_2=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_2 --cidr-block 192.168.1.32/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet02},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
```

> **Important**
>
> Par défaut, vos nouveaux sous-réseaux sont implicitement associés à la table de routage principale de votre VPC. Cette table de routage permet la communication entre toutes les ressources déployées dans le VPC. Toutefois, elle n'autorise pas la communication avec des ressources dont les adresses IP se trouvent en dehors des blocs CIDR associés à votre VPC. Vous pouvez associer votre propre table de routage à vos sous-réseaux pour modifier ce comportement. Pour plus

d'informations, consultez Tables de routage de sous-réseau dans le Guide de l'utilisateur Amazon VPC.

2. Affichez les sous-réseaux actuels dans votre VPC.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" \
    --query 'Subnets[*].{SubnetId: SubnetId,AvailabilityZone:
 AvailabilityZone,CidrBlock: CidrBlock}' \
    --output table
```

L'exemple qui suit illustre un résultat.

```
-------------------------------------------------------------------
|                          DescribeSubnets                        |
+------------------+--------------------+--------------------------+
| AvailabilityZone |     CidrBlock      |         SubnetId         |
+------------------+--------------------+--------------------------+
|  us-west-2d      |  192.168.0.0/27    |     subnet-example1      |
|  us-west-2a      |  192.168.0.32/27   |     subnet-example2      |
|  us-west-2a      |  192.168.0.64/27   |     subnet-example3      |
|  us-west-2d      |  192.168.0.96/27   |     subnet-example4      |
|  us-west-2a      |  192.168.1.0/27    |     subnet-example5      |
|  us-west-2d      |  192.168.1.32/27   |     subnet-example6      |
+------------------+--------------------+--------------------------+
```

Vous pouvez voir que les sous-réseaux se trouvant dans le bloc CIDR `192.168.1.0` que vous avez créé se trouvent dans les mêmes zones de disponibilité que les sous-réseaux dans le bloc d'adresse CIDR `192.168.0.0`.

Étape 3 : Configuration des ressources Kubernetes

Pour configurer les ressources Kubernetes

1. Définissez la variable d'environnement `AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG` sur `true` dans le `aws-node DaemonSet`.

```
kubectl set env daemonset aws-node -n kube-system
 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG=true
```

2. Récupérez l'ID de votre groupe de sécurité de cluster et stockez-le dans une variable pour une utilisation ultérieure. Amazon EKS crée automatiquement ce groupe de sécurité lorsque vous créez votre cluster.

```
cluster_security_group_id=$(aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId --output text)
```

3. Créez une ressource personnalisée `ENIConfig` pour chaque sous-réseau dans lequel vous souhaitez déployer les Pods.

   a. Créez un fichier unique pour chaque configuration d'interface réseau.

   Les commandes suivantes créent des fichiers `ENIConfig` distincts pour les deux sous-réseaux créés à l'étape précédente. La valeur pour `name` doit être unique. Le nom est le même que la zone de disponibilité dans laquelle se trouve le sous-réseau. Le groupe de sécurité de cluster est affecté au `ENIConfig`.

```
cat >$az_1.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_1
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_1
EOF
```

```
cat >$az_2.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_2
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_2
EOF
```

Pour un cluster de production, vous pouvez apporter les modifications suivantes aux commandes précédentes :

- Remplacez *$cluster_security_group_id* par l'ID d'un groupe de sécurité existant que vous souhaitez utiliser pour chaque `ENIConfig`.
- Nous vous recommandons de nommer vos `ENIConfigs` de la même façon que la zone de disponibilité pour laquelle vous allez utiliser le `ENIConfig`, chaque fois que c'est possible. Vous devrez peut-être utiliser pour vos `ENIConfigs` des noms différents de ceux des zones de disponibilité pour diverses raisons. Par exemple, si vous avez plus de deux sous-réseaux dans la même zone de disponibilité et que vous souhaitez les utiliser tous les deux avec une mise en réseau personnalisée, vous avez besoin de plusieurs `ENIConfigs` pour la même zone de disponibilité. Puisque chaque `ENIConfig` nécessite un nom unique, vous ne pouvez nommer plus d'un de vos `ENIConfigs` en utilisant le nom de la zone de disponibilité.

  Si vos noms `ENIConfig` ne sont pas tous les mêmes que les noms de zone de disponibilité, remplacez *$az_1* et *$az_2* par vos propres noms dans les commandes précédentes et annotez vos nœuds avec le `ENIConfig` plus tard dans ce didacticiel.

> **Note**
>
> Si vous ne spécifiez pas de groupe de sécurité valide à utiliser avec un cluster de production et que vous utilisez :
>
> - la version `1.8.0` ou une version ultérieure du Amazon VPC CNI plugin for Kubernetes, alors les groupes de sécurité associés à l'interface réseau Elastic principale du nœud sont utilisés.
> - une version du Amazon VPC CNI plugin for Kubernetes antérieure à la version `1.8.0`, le groupe de sécurité par défaut du VPC est attribué aux interfaces réseau secondaires.

> **Important**
>
> - `AWS_VPC_K8S_CNI_EXTERNALSNAT=false` est un paramètre par défaut dans la configuration du plugin CNI Amazon VPC pour Kubernetes. Si vous utilisez

le paramètre par défaut, le trafic destiné aux adresses IP qui ne se trouvent pas dans l'un des blocs CIDR associés à votre VPC utilise les groupes de sécurité et les sous-réseaux de l'interface réseau principale de votre nœud. Les sous-réseaux et les groupes de sécurité définis dans vos `ENIConfigs` qui sont utilisés pour créer des interfaces réseau secondaires ne sont pas utilisées pour ce trafic. Pour plus d'informations sur ce paramètre, consultez SNAT pour Pods.

- Si vous utilisez également des groupes de sécurité pour des Pods, le groupe de sécurité spécifié dans un `SecurityGroupPolicy` est utilisé à la place du groupe de sécurité spécifié dans les `ENIConfigs`. Pour plus d'informations, consultez Groupes de sécurité pour Pods.

b. Appliquez chaque fichier de ressource personnalisé que vous avez créé à votre cluster à l'aide des commandes suivantes.

```
kubectl apply -f $az_1.yaml
kubectl apply -f $az_2.yaml
```

4. Confirmez que vos `ENIConfigs` ont été créés.

```
kubectl get ENIConfigs
```

L'exemple qui suit illustre un résultat.

```
NAME         AGE
us-west-2a   117s
us-west-2d   105s
```

5. Si vous activez la mise en réseau personnalisée sur un cluster de production et que vous avez nommé vos `ENIConfigs` autrement qu'avec le nom de la zone de disponibilité pour laquelle vous les utilisez, passez à l'étape suivante pour déployer des nœuds Amazon EC2.

   Permettez à Kubernetes d'appliquer automatiquement le `ENIConfig` pour une zone de disponibilité pour tous les nouveaux nœuds Amazon EC2 créés dans votre cluster.

   1. Pour le cluster test de ce didacticiel, passez à l'étape suivante.

      Pour un cluster de production, vérifiez si une annotation avec clé `k8s.amazonaws.com/eniConfig` pour la variable d'environnement `ENI_CONFIG_ANNOTATION_DEF` existe dans la spécification du conteneur pour le `aws-node DaemonSet`.

```
kubectl describe daemonset aws-node -n kube-system | grep
 ENI_CONFIG_ANNOTATION_DEF
```

Si la sortie est renvoyée, l'annotation existe. Si aucune sortie n'est renvoyée, la variable n'est pas définie. Pour un cluster de production, vous pouvez utiliser ce paramètre ou le paramètre de l'étape suivante. Si vous utilisez ce paramètre, il remplace le paramètre de l'étape suivante. Dans ce didacticiel, le paramètre de l'étape suivante est utilisé.

2. Mettez à jour votre `aws-node` `DaemonSet` pour appliquer automatiquement le `ENIConfig` pour une zone de disponibilité pour tous les nouveaux nœuds Amazon EC2 créés dans votre cluster.

```
kubectl set env daemonset aws-node -n kube-system
 ENI_CONFIG_LABEL_DEF=topology.kubernetes.io/zone
```

Étape 4 : déploiement de nœuds Amazon EC2

Pour déployer des nœuds Amazon EC2

1. Créez un rôle IAM de nœud.

   a. Exécutez la commande suivante pour créer un fichier JSON de politique d'approbation IAM.

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

b. Exécutez la commande suivante pour définir une variable pour votre nom de rôle. Vous pouvez remplacer *myCustomNetworkingAmazonEKSNodeRole* par n'importe quel nom que vous choisissez.

```
export node_role_name=myCustomNetworkingAmazonEKSNodeRole
```

c. Créez le rôle IAM et stockez son Amazon Resource Name (ARN) renvoyé dans une variable pour une utilisation ultérieure.

```
node_role_arn=$(aws iam create-role --role-name $node_role_name --assume-role-
policy-document file://"node-role-trust-relationship.json" \
    --query Role.Arn --output text)
```

d. Attachez trois politiques gérées IAM requises au rôle IAM.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Pour simplifier ce tutoriel, la politique AmazonEKS_CNI_Policy est attachée au nœud du rôle IAM. Toutefois, dans un cluster de production, nous recommandons de rattacher la politique à un rôle IAM distinct utilisé uniquement avec le Amazon VPC CNI plugin for Kubernetes. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

2. Créez l'un des types de groupes de nœuds suivants. Pour déterminer le type d'instance que vous souhaitez déployer, consultez Choix d'un type d'instance Amazon EC2. Pour ce didacticiel, terminez l'option Gérée, Sans modèle de lancement ou avec un modèle de lancement sans ID d'AMI spécifié. Si vous comptez utiliser le groupe de nœuds pour les charges de travail de

production, nous vous recommandons de vous familiariser avec toutes les options de groupe de nœuds gérés et autogérés avant de déployer le groupe de nœuds.

- Géré : déployez votre groupe de nœuds à l'aide de l'une des options suivantes :
  - Sans modèle de lancement ou avec un modèle de lancement sans ID d'AMI spécifié : exécutez la commande suivante. Pour ce didacticiel, utilisez *example values*. Pour un groupe de nœuds de production, remplacez toutes les *example values* par vos propres valeurs. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères.

    ```
    aws eks create-nodegroup --cluster-name $cluster_name --nodegroup-name my-nodegroup \
        --subnets $subnet_id_1 $subnet_id_2 --instance-types t3.medium --node-role $node_role_arn
    ```

  - Avec un modèle de lancement avec un ID AMI spécifié
    1. Déterminez le nombre maximal de Pods recommandé par Amazon EKS pour vos nœuds. Suivez les instructions de la section Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2, en ajoutant **--cni-custom-networking-enabled** à l'étape 3 de cette rubrique. Notez la sortie pour l'utiliser lors de l'étape suivante.
    2. Dans votre modèle de lancement, spécifiez un ID d'AMI optimisé pour Amazon EKS ou une AMI personnalisée créée à partir de l'AMI optimisée pour Amazon EKS, puis déployez le groupe de nœuds avec un modèle de lancement et fournissez les données utilisateur suivantes dans le modèle de lancement. Ces données utilisateur transmettent des arguments dans le fichier `bootstrap.sh`. Pour plus d'informations sur le fichier d'amorçage, consultez bootstrap.sh sur GitHub. Vous pouvez remplacer *20* par la valeur de l'étape précédente (recommandé) ou par votre propre valeur.

       ```
       /etc/eks/bootstrap.sh my-cluster --use-max-pods false --kubelet-extra-args '--max-pods=20'
       ```

       Si vous avez créé une AMI personnalisée qui n'est pas créée à partir de l'AMI optimisée pour Amazon EKS, vous devez créer vous-même la configuration.
- Autogéré

1. Déterminez le nombre maximal de Pods recommandé par Amazon EKS pour vos nœuds. Suivez les instructions de la section Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2, en ajoutant **`--cni-custom-networking-enabled`** à l'étape 3 de cette rubrique. Notez la sortie pour l'utiliser lors de l'étape suivante.
2. Déployez le groupe de nœuds à l'aide des instructions contenues dans Lancement de nœuds Amazon Linux autogérés. Spécifiez le texte suivant pour le BootstrapArgumentsparamètre. Vous pouvez remplacer *`20`* par la valeur de l'étape précédente (recommandé) ou par votre propre valeur.

```
--use-max-pods false --kubelet-extra-args '--max-pods=20'
```

> **Note**
>
> Si vous souhaitez que les nœuds d'un cluster de production prennent en charge un nombre significativement plus élevé de Pods, exécutez à nouveau le script dans Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2. Ajoutez également l'option **`--cni-prefix-delegation-enabled`** à la commande. Par exemple, *`110`* est renvoyé pour un type d'instance `m5.large`. Pour obtenir des instructions sur la façon d'activer cette capacité, consultez Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2. Vous pouvez utiliser cette capacité avec une mise en réseau personnalisée.

La création d'un groupe de nœuds dure plusieurs minutes. Vous pouvez vérifier l'état de la création d'un groupe de nœuds géré à l'aide de la commande suivante.

```
aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-nodegroup --query nodegroup.status --output text
```

Ne passez pas à l'étape suivante tant que la sortie retournée n'est pas `ACTIVE`.

3. Pour le didacticiel, vous pouvez ignorer cette étape.

   Pour un cluster de production, si vous n'avez pas nommé vos `ENIConfigs` de la même manière que la zone de disponibilité pour laquelle vous les utilisez, vous devez annoter vos nœuds avec le nom `ENIConfig` qui doit être utilisé avec le nœud. Cette étape n'est pas nécessaire si vous

n'avez qu'un seul sous-réseau dans chaque zone de disponibilité et que vous avez nommé vos `ENIConfigs` avec les mêmes noms que vos zones de disponibilité. Cela est dû au fait que le Amazon VPC CNI plugin for Kubernetes associe automatiquement le bon `ENIConfig` avec le bon nœud si vous l'y avez autorisé lors d'une étape précédente.

a. Obtenez la liste des nœuds de votre cluster.

```
kubectl get nodes
```

L'exemple qui suit illustre un résultat.

```
NAME                                           STATUS   ROLES    AGE     VERSION
ip-192-168-0-126.us-west-2.compute.internal   Ready    <none>   8m49s
 v1.22.9-eks-810597c
ip-192-168-0-92.us-west-2.compute.internal    Ready    <none>   8m34s
 v1.22.9-eks-810597c
```

b. Déterminez dans quelle zone de disponibilité se trouve chaque nœud. Exécutez la commande suivante pour chaque nœud qui a été renvoyé lors de l'étape précédente.

```
aws ec2 describe-instances --filters Name=network-interface.private-dns-
name,Values=ip-192-168-0-126.us-west-2.compute.internal \
--query 'Reservations[].Instances[].{AvailabilityZone:
 Placement.AvailabilityZone, SubnetId: SubnetId}'
```

L'exemple qui suit illustre un résultat.

```
[
    {
        "AvailabilityZone": "us-west-2d",
        "SubnetId": "subnet-Example5"
    }
]
```

c. Annotez chaque nœud avec le `ENIConfig` que vous avez créé pour l'ID de sous-réseau et la zone de disponibilité. Vous ne pouvez annoter un nœud qu'avec un seul `ENIConfig`, bien que plusieurs nœuds puissent être annotés avec le même `ENIConfig`. Remplacez les *`example values`* par vos propres valeurs.

```
kubectl annotate node ip-192-168-0-126.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName1
kubectl annotate node ip-192-168-0-92.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName2
```

4. Si vous aviez des nœuds dans un cluster de production avec des Pods en cours d'exécution avant de passer à l'utilisation de la fonctionnalité de mise en réseau personnalisée, exécutez les tâches suivantes :

   a. Assurez-vous que vous disposez de nœuds disponibles utilisant la fonctionnalité de mise en réseau personnalisée.

   b. Bouclez et videz les nœuds pour arrêter gracieusement les Pods. Pour plus d'informations, consultez Drainer un nœud en toute sécurité dans la documentation Kubernetes.

   c. Mettez fin aux nœuds. Si les nœuds se trouvent dans un groupe de nœuds géré existant, vous pouvez supprimer le groupe de nœuds. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée :

      - Remplacez *my-cluster* par le nom de votre cluster.
      - Remplacez *my-nodegroup* par le nom de votre groupe de nœuds.

```
aws eks delete-nodegroup --cluster-name my-cluster --nodegroup-name my-
nodegroup
```

   Seuls les nouveaux nœuds enregistrés auprès du label `k8s.amazonaws.com/eniConfig` utilisent la fonctionnalité de réseaux personnalisés.

5. Confirmez que les Pods se voient attribuer une adresse IP à partir d'un bloc CIDR associé à l'un des sous-réseaux que vous avez créés lors d'une étape précédente.

```
kubectl get pods -A -o wide
```

   L'exemple qui suit illustre un résultat.

```
NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE     IP
           NODE                                            NOMINATED NODE   READINESS
 GATES
```

```
kube-system   aws-node-2rkn4             1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   aws-node-k96wp             1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
kube-system   coredns-657694c6f4-smcgr   1/1     Running   0          56m
 192.168.1.23    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   coredns-657694c6f4-stwv9   1/1     Running   0          56m
 192.168.1.28    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-jgshq           1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-wx9vk           1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
```

Vous pouvez voir que les Pods `coredns` sont attribuées à des adresses IP à partir du bloc d'adresse CIDR `192.168.1.0` que vous avez ajouté à votre VPC. Sans mise en réseau personnalisée, des adresses auraient été attribuées à partir du bloc CIDR `192.168.0.0`, car il s'agissait du seul bloc CIDR initialement associé au VPC.

Si un `spec` Pod's contient `hostNetwork=true`, l'adresse IP principale du nœud lui est attribuée. Aucune adresse ne lui est attribuée à partir des sous-réseaux que vous avez ajoutés. Par défaut, cette valeur indique `false`. Cette valeur est définie sur `true` pour le `kube-proxy` et les Pods du Amazon VPC CNI plugin for Kubernetes (`aws-node`) qui s'exécutent sur votre cluster. C'est pourquoi le `kube-proxy` et les Pods du plugin `aws-node` n'ont pas d'adresses `192.168.1.x` attribuées dans la sortie précédente. Pour plus d'informations sur un Pod's `hostNetwork` paramètre, voir PodSpec v1 core dans la référence de l'KubernetesAPI.

Étape 5 : suppression des ressources du didacticiel

Une fois le didacticiel terminé, nous vous recommandons de supprimer les ressources que vous avez créées. Vous pouvez ensuite ajuster les étapes pour activer la mise en réseau personnalisée pour un cluster de production.

Pour supprimer les ressources du didacticiel

1. Si le groupe de nœuds que vous avez créé n'était qu'à des fins de test, supprimez-le.

   ```
   aws eks delete-nodegroup --cluster-name $cluster_name --nodegroup-name my-nodegroup
   ```

   Même une fois que le AWS CLI résultat indique que le cluster est supprimé, le processus de suppression peut ne pas être terminé. Le processus de suppression prend quelques minutes. Vérifiez qu'il est terminé en exécutant la commande suivante.

   ```
   aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-
   nodegroup --query nodegroup.status --output text
   ```

   Ne continuez pas tant que la sortie renvoyée n'est pas similaire à la sortie suivante.

   ```
   An error occurred (ResourceNotFoundException) when calling the DescribeNodegroup
    operation: No node group found for name: my-nodegroup.
   ```

2. Si le groupe de nœuds que vous avez créé n'était qu'à des fins de test, supprimez le rôle IAM du nœud.

   a. Détachez les politiques du rôle.

      ```
      aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
      policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
      aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
      policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
      aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
      policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
      ```

   b. Supprimez le rôle.

      ```
      aws iam delete-role --role-name myCustomNetworkingAmazonEKSNodeRole
      ```

3. Supprimez le cluster.

   ```
   aws eks delete-cluster --name $cluster_name
   ```

   Vérifiez que le cluster est supprimé avec la commande suivante.

```
aws eks describe-cluster --name $cluster_name --query cluster.status --output text
```

Lorsqu'une sortie similaire à la suivante est renvoyée, le cluster est correctement supprimé.

```
An error occurred (ResourceNotFoundException) when calling the DescribeCluster
 operation: No cluster found for name: my-cluster.
```

4. Supprimez le rôle IAM de cluster.

   a. Détachez les politiques du rôle.

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSClusterRole
    --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
   ```

   b. Supprimez le rôle.

   ```
   aws iam delete-role --role-name myCustomNetworkingAmazonEKSClusterRole
   ```

5. Supprimez les sous-réseaux que vous avez créés à l'étape précédente.

   ```
   aws ec2 delete-subnet --subnet-id $new_subnet_id_1
   aws ec2 delete-subnet --subnet-id $new_subnet_id_2
   ```

6. Supprimez le VPC que vous avez créé.

   ```
   aws cloudformation delete-stack --stack-name my-eks-custom-networking-vpc
   ```

Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2

Chaque instance Amazon EC2 prend en charge un nombre maximum d'interfaces réseau élastiques et un nombre maximum d'adresses IP pouvant être attribuées à chaque interface réseau. Chaque nœud a besoin d'une adresse IP pour chaque interface réseau. Toutes les autres adresses IP disponibles peuvent être attribuées aux `Pods`. Chaque `Pod` a besoin de sa propre adresse IP. Par conséquent, vous pouvez avoir des nœuds qui disposent de ressources de calcul et de mémoire disponibles, mais qui ne peuvent pas accueillir d'autres `Pods` parce que le nœud n'a plus d'adresses IP à attribuer à `Pods`.

Dans cette rubrique, vous allez apprendre à augmenter de manière significative le nombre d'adresses IP que les nœuds peuvent attribuer à `Pods` en attribuant des préfixes IP, plutôt que d'attribuer des

adresses IP secondaires individuelles à vos nœuds. Chaque préfixe inclut plusieurs adresses IP. Si vous ne configurez pas votre cluster pour l'attribution de préfixes IP, votre cluster doit effectuer davantage d'appels à l'interface de programmation d'applications (API) Amazon EC2 afin de configurer les interfaces réseau et les adresses IP nécessaires à la connectivité Pod. Au fur et à mesure que la taille des clusters augmente, la fréquence de ces appels d'API peut entraîner des temps de lancement de Pod et d'instances plus longs. Cela entraîne des retards de mise à l'échelle pour répondre à la demande d'applications volumineuses et épineuses, et ajoute des coûts et des frais généraux de gestion, car vous devez allouer des clusters et des VPC supplémentaires pour répondre aux exigences de mise à l'échelle. Pour plus d'informations, consultez la section Seuils Kubernetes d'évolutivité activés GitHub.

Considérations

- Chaque type d'instance Amazon EC2 prend en charge un nombre maximum de Pods. Si votre groupe de nœuds gérés est composé de plusieurs types d'instance, le plus petit nombre de Pods maximum pour une instance dans le cluster est appliqué à tous les nœuds du cluster.
- Par défaut, le nombre maximum de `Pods` que vous pouvez exécuter sur un nœud est de 110, mais vous pouvez modifier ce nombre. Si vous modifiez le numéro et que vous disposez d'un groupe de nœuds géré existant, la prochaine mise à jour d'AMI ou de modèle de lancement de votre groupe de nœuds entraînera la création de nouveaux nœuds avec la valeur modifiée.
- Lorsque vous passez de l'attribution d'adresses IP à l'attribution de préfixes IP, nous vous recommandons de créer de nouveaux groupes de nœuds afin d'augmenter le nombre d'adresses IP disponibles, plutôt que de remplacer progressivement les nœuds existants. L'exécution Pods sur un nœud auquel des adresses IP et des préfixes ont été attribués peut entraîner des incohérences dans la capacité des adresses IP annoncée, ce qui a un impact sur les charges de travail futures sur le nœud. Pour connaître la méthode recommandée pour effectuer la transition, consultez Remplacer tous les nœuds lors de la migration du mode IP secondaire vers le mode Délégation de préfixe ou vice versa dans le guide des meilleures pratiques Amazon EKS.
- Pour les clusters dotés de nœuds Linux uniquement.
  - Une fois que vous avez configuré le module complémentaire pour attribuer des préfixes aux interfaces réseau, vous ne pouvez pas rétrograder votre module complémentaire Amazon VPC CNI plugin for Kubernetes vers une version inférieure à `1.9.0` (ou `1.10.1`) sans supprimer tous les nœuds de tous les groupes de nœuds de votre cluster.
  - Si vous utilisez également des groupes de sécurité pour Pods, avec `POD_SECURITY_GROUP_ENFORCING_MODE` = `standard` et `AWS_VPC_K8S_CNI_EXTERNALSNAT` =`false`, lorsque vos Pods communiquent avec des points

de terminaison extérieurs à votre VPC, les groupes de sécurité du nœud sont utilisés, plutôt que les groupes de sécurité que vous avez attribués à vos Pods.

Si vous utilisez également des groupes de sécurité pour Pods, avec `POD_SECURITY_GROUP_ENFORCING_MODE=strict`, lorsque vos `Pods` communiquent avec des points de terminaison extérieurs à votre VPC, les groupes de sécurité de `Pod's` sont utilisés.

Prérequis

- Un cluster existant. Pour en déployer un, consultez Création d'un cluster Amazon EKS.
- Les sous-réseaux dans lesquels se trouvent vos nœuds Amazon EKS doivent comporter suffisamment de blocs de routage inter-domaines sans classe (CIDR) contigus `/28` (pour les clusters `IPv4`) ou `/80` (pour les clusters `IPv6`). Vous ne pouvez avoir que des nœuds Linux dans un cluster `IPv6`. L'utilisation de préfixes IP peut échouer si les adresses IP sont dispersées dans le CIDR du sous-réseau. Nous vous recommandons la procédure suivante :
  - L'utilisation d'une réservation CIDR de sous-réseau afin que, même si des adresses IP comprises dans la plage réservée sont encore utilisées, elles ne soient pas réattribuées lors de leur libération. Cela permet de s'assurer que les préfixes sont disponibles pour l'allocation sans segmentation.
  - Utilisez de nouveaux sous-réseaux spécifiquement utilisés pour exécuter les charges de travail auxquelles les préfixes IP sont attribués. Les charges de travail Windows et Linux peuvent être exécutées dans le même sous-réseau lors de l'attribution des préfixes IP.
- Pour attribuer des préfixes IP à vos nœuds, ceux-ci doivent être basés sur AWS Nitro. Les instances qui ne sont pas basées sur Nitro continuent d'allouer des adresses IP secondaires individuelles, mais ont un nombre nettement inférieur d'adresses IP à attribuer aux Pods par rapport aux instances Nitro-based.
- Pour les clusters avec des nœuds Linux uniquement : si votre cluster est configuré pour la famille `IPv4`, la version `1.9.0` ou une version ultérieure du module complémentaire Amazon VPC CNI plugin for Kubernetes doit être installée. Vous pouvez vérifier votre version actuelle à l'aide de la commande suivante.

```
kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/"
 -f 2
```

Si votre cluster est configuré pour la famille `IPv6`, la version `1.10.1` du module complémentaire doit être installée. Si la version de votre plugin est antérieure aux versions requises, vous devez la mettre à jour. Pour en savoir plus, consultez les sections de mise à jour de Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

- Pour les clusters avec des nœuds Windows uniquement
  - La version de votre cluster et de sa plateforme doit être égale ou ultérieure aux versions indiquées dans le tableau suivant. Pour mettre à jour la version de votre cluster, consultez Mise à jour d'une version Kubernetes de cluster Amazon EKS. Si votre cluster ne dispose pas de la version minimale de la plateforme, vous ne pouvez pas attribuer de préfixes IP à vos nœuds tant qu'Amazon EKS n'a pas mis à jour votre version de la plateforme.

Version de Kubernetes	Version de la plateforme
`1.27`	`eks.3`
`1.26`	`eks.4`
`1.25`	`eks.5`

  Vous pouvez vérifier votre version actuelle de Kubernetes et de la plateforme en remplaçant *`my-cluster`* dans la commande suivante par le nom de votre cluster et en exécutant la commande modifiée : **`aws eks describe-cluster --name my-cluster --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'`**.
  - Prise en charge de Windows activée pour votre cluster. Pour plus d'informations, consultez Activation de la prise en charge de Windows pour votre cluster Amazon EKS.

Pour augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2

1. Configurez votre cluster pour attribuer des préfixes d'adresses IP aux nœuds. Effectuez la procédure dans l'onglet correspondant au système d'exploitation de votre nœud.

   Linux

   1. Activez le paramètre pour affecter des préfixes aux interfaces réseau pour le processus CNI Amazon VPC `DaemonSet`. Lorsque vous déployez un cluster de version `1.21` ou ultérieure, la version `1.10.1` ou ultérieure du module complémentaire Amazon VPC CNI

plugin for Kubernetes est déployée avec lui. Si vous avez créé le cluster avec la famille `IPv6`, ce paramètre a été réglé sur `true` par défaut. Si vous avez créé le cluster avec la famille `IPv4`, ce paramètre a été réglé sur `false` par défaut.

```
kubectl set env daemonset aws-node -n kube-system
 ENABLE_PREFIX_DELEGATION=true
```

> **Important**
>
> Même si votre sous-réseau a des adresses IP disponibles, si le sous-réseau n'a pas de blocs contigus `/28` disponibles, vous verrez le message d'erreur suivant dans les journaux Amazon VPC CNI plugin for Kubernetes.
>
> ```
> InsufficientCidrBlocks: The specified subnet does not have enough free
>  cidr blocks to satisfy the request
> ```
>
> Cela peut se produire en raison de la fragmentation des adresses IP secondaires existantes réparties sur un sous-réseau. Pour résoudre cette erreur, créez un nouveau sous-réseau et lancez des Pods là-bas, ou utilisez une réservation CIDR de sous-réseau Amazon EC2 pour réserver de l'espace dans un sous-réseau à utiliser avec l'affectation de préfixe. Pour plus d'informations, consultez la section Réservations CIDR de sous-réseau dans le Guide de l'utilisateur Amazon VPC.

2. Si vous prévoyez de déployer un groupe de nœuds gérés sans modèle de lancement ou avec un modèle de lancement dans lequel vous n'avez pas spécifié d'ID AMI, et si vous utilisez une version du Amazon VPC CNI plugin for Kubernetes égale ou supérieure aux versions répertoriées dans les prérequis, passez à l'étape suivante. Les groupes de nœuds gérés calculent automatiquement le nombre maximal de Pods pour vous.

   Si vous déployez un groupe de nœuds autogérés ou un groupe de nœuds gérés avec un modèle de lancement dans lequel vous avez spécifié un ID d'AMI, vous devez déterminer le nombre maximal de Pods recommandés par Amazon EKS pour vos nœuds. Suivez les instructions de la section Nombre maximal de Pods recommandé par Amazon EKS pour chaque type d'instance Amazon EC2, en ajoutant **`--cni-prefix-delegation-enabled`** à l'étape 3. Notez la sortie pour une utilisation ultérieure.

> **Important**
>
> Les groupes de nœuds gérés appliquent un nombre maximal sur la valeur de `maxPods`. Pour les instances avec moins de 30 vCPUs, le nombre maximum est 110 et pour toutes les autres instances, le nombre maximum est 250. Ce nombre maximal est appliqué que la délégation du préfixe soit activée ou non.

3. Si vous utilisez un cluster `1.21` ou version ultérieure configuré pour `IPv6`, passez directement à l'étape suivante.

   Spécifiez les paramètres dans l'une des options suivantes. Pour déterminer l'option qui vous convient le mieux et la valeur à fournir, consultez `WARM_PREFIX_TARGET`, `WARM_IP_TARGET` et `MINIMUM_IP_TARGET` sur GitHub.

   Vous pouvez remplacer les *example values* par une valeur supérieure à zéro.

   - `WARM_PREFIX_TARGET`

     ```
     kubectl set env ds aws-node -n kube-system WARM_PREFIX_TARGET=1
     ```

   - `WARM_IP_TARGET` ou `MINIMUM_IP_TARGET` : si l'une ou l'autre des valeurs est définie, elle remplace toute valeur définie pour `WARM_PREFIX_TARGET`.

     ```
     kubectl set env ds aws-node -n kube-system WARM_IP_TARGET=5
     ```

     ```
     kubectl set env ds aws-node -n kube-system MINIMUM_IP_TARGET=2
     ```

4. Créez l'un des types de groupes de nœuds suivants avec au moins un type d'instance Amazon EC2 Nitro Amazon Linux 2. Pour obtenir la liste des types d'instances Nitro, consultez la section Instances créées sur le système Nitro dans le guide de l'utilisateur Amazon EC2. Cette capacité n'est pas prise en charge sur Windows. Pour les options qui incluent *110*, remplacez-le par la valeur de l'étape 3 (recommandée) ou par votre propre valeur.

   - Gestion automatique : déploie le groupe de nœuds à l'aide des instructions de Lancement de nœuds Amazon Linux autogérés. Spécifiez le texte suivant pour le BootstrapArgumentsparamètre.

     ```
     --use-max-pods false --kubelet-extra-args '--max-pods=110'
     ```

Si vous utilisez `eksctl` pour créer le groupe de nœuds, vous pouvez utiliser la commande suivante.

```
eksctl create nodegroup --cluster my-cluster --managed=false --max-pods-per-node 110
```

- Géré : déployez votre groupe de nœuds à l'aide de l'une des options suivantes :
  - Sans modèle de lancement ou avec un modèle de lancement sans ID d'AMI spécifié : exécutez la procédure dans Création d'un groupe de nœuds gérés. Les groupes de nœuds gérés calculent automatiquement pour vous la valeur `max-pods` recommandée par Amazon EKS.
  - Avec un modèle de lancement avec un ID d'AMI spécifié : dans votre modèle de lancement, spécifiez un ID d'AMI optimisé pour Amazon EKS ou une AMI personnalisée créée à partir de l'AMI optimisée pour Amazon EKS, puis Déployez le groupe de nœuds avec un modèle de lancement et fournissez les données utilisateur suivantes dans le modèle de lancement. Ces données utilisateur transmettent des arguments dans le fichier `bootstrap.sh`. Pour plus d'informations sur le fichier d'amorçage, consultez bootstrap.sh sur GitHub.

    ```
    /etc/eks/bootstrap.sh my-cluster \
      --use-max-pods false \
      --kubelet-extra-args '--max-pods=110'
    ```

    Si vous utilisez `eksctl` pour créer le groupe de nœuds, vous pouvez utiliser la commande suivante.

    ```
    eksctl create nodegroup --cluster my-cluster --max-pods-per-node 110
    ```

    Si vous avez créé une AMI personnalisée qui n'est pas créée à partir de l'AMI optimisée pour Amazon EKS, vous devez créer vous-même la configuration.

> **Note**
>
> Si vous souhaitez également affecter des adresses IP à des Pods à partir d'un sous-réseau différent de celui de l'instance, vous devez activer cette

fonctionnalité dans cette étape. Pour plus d'informations, consultez Mise en réseau personnalisée pour les pods.

Windows

1. Activez l'attribution de préfixes IP.

   a. Ouvrez le `amazon-vpc-cni` `ConfigMap` pour le modifier.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Ajoutez les lignes suivantes à la section `data`.

   ```
     enable-windows-prefix-delegation: "true"
   ```

   c. Enregistrez le fichier et fermez l'éditeur.

   d. Confirmez que la ligne a été ajoutée à la `ConfigMap`.

   ```
   kubectl get configmap -n kube-system amazon-vpc-cni -o
    "jsonpath={.data.enable-windows-prefix-delegation}"
   ```

   Si la sortie renvoyée n'est pas `true`, il se peut qu'il y ait eu une erreur. Essayez à nouveau d'effectuer l'étape.

   Important

   Même si votre sous-réseau a des adresses IP disponibles, si le sous-réseau n'a pas de blocs contigus `/28` disponibles, vous verrez le message d'erreur suivant dans les événements du nœud.

   ```
   "failed to allocate a private IP/Prefix address:
    InsufficientCidrBlocks: The specified subnet does not have enough
    free cidr blocks to satisfy the request"
   ```

   Cela peut se produire en raison de la fragmentation des adresses IP secondaires existantes réparties sur un sous-réseau. Pour résoudre cette erreur, créez un nouveau sous-réseau et lancez des Pods là-bas, ou utilisez une réservation CIDR de sous-réseau Amazon EC2 pour réserver de

l'espace dans un sous-réseau à utiliser avec l'affectation de préfixe. Pour plus d'informations, consultez la section Réservations CIDR de sous-réseau dans le Guide de l'utilisateur Amazon VPC.

2. (Facultatif) Spécifiez une configuration supplémentaire pour contrôler le comportement de pré-échelonnement et d'échelonnement dynamique de votre cluster. Pour plus d'informations, consultez la section Options de configuration avec le mode de délégation de préfixes activé Windows. GitHub

   a. Ouvrez le `amazon-vpc-cni ConfigMap` pour le modifier.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Remplacez le *example values* par une valeur supérieure à zéro et ajoutez les entrées dont vous avez besoin dans la section `data` du `ConfigMap`. Si vous définissez une valeur pour `warm-ip-target` ou `minimum-ip-target`, la valeur remplace toute valeur définie pour `warm-prefix-target`.

   ```
   warm-prefix-target: "1"
   warm-ip-target: "5"
   minimum-ip-target: "2"
   ```

   c. Enregistrez le fichier et fermez l'éditeur.

3. Créez des groupes de nœuds Windows avec au moins un type d'instance Amazon EC2 Nitro. Pour obtenir la Nitro liste des types d'instances, consultez la section Instances créées sur le Nitro système dans le guide de l'utilisateur Amazon EC2. Par défaut, le nombre maximum de Pods que vous pouvez déployer sur un nœud est de 110. Si vous souhaitez augmenter ou diminuer ce nombre, indiquez ce qui suit dans les données utilisateur de la configuration d'amorçage. Remplacez *max-pods-quantity* par votre valeur maximale de pods.

   ```
   -KubeletExtraArgs '--max-pods=max-pods-quantity'
   ```

   Si vous déployez des groupes de nœuds gérés, cette configuration doit être ajoutée au modèle de lancement. Pour plus d'informations, consultez Personnalisation des nœuds gérés avec des modèles de lancement. Pour plus d'informations sur les paramètres de configuration du script d'amorçage de Windows veuillez consulter Paramètres de configuration du script d'amorçage.

2. Une fois que vos nœuds sont déployés, affichez les nœuds de votre cluster.

```
kubectl get nodes
```

L'exemple qui suit illustre un résultat.

```
NAME                                               STATUS     ROLES    AGE   VERSION
ip-192-168-22-103.region-code.compute.internal   Ready      <none>   19m   v1.XX.X-eks-6b7464
ip-192-168-97-94.region-code.compute.internal    Ready      <none>   19m   v1.XX.X-eks-6b7464
```

3. Décrivez l'un des nœuds pour déterminer la valeur de `max-pods` pour le nœud et le nombre d'adresses IP disponibles. Remplacez *192.168.30.193* par l'adresse `IPv4` dans le nom de l'un de vos nœuds renvoyés dans la sortie précédente.

```
kubectl describe node ip-192-168-30-193.region-code.compute.internal | grep 'pods\|PrivateIPv4Address'
```

L'exemple qui suit illustre un résultat.

```
pods:                                       110
vpc.amazonaws.com/PrivateIPv4Address:  144
```

Dans la sortie précédente, `110` est le nombre maximum de Pods que Kubernetes déploiera sur le nœud, même si *144* adresses IP sont disponibles.

Groupes de sécurité pour Pods

Les groupes de sécurité pour les Pods intègrent des groupes de sécurité Amazon EC2 avec des Pods Kubernetes. Vous pouvez utiliser les groupes de sécurité Amazon EC2 pour définir des règles qui autorisent le trafic réseau entrant et sortant à destination et en provenance des Pods que vous déployez vers des nœuds s'exécutant sur de nombreux types d'instances Amazon EC2 et Fargate. Pour obtenir une explication détaillée de cette capacité, consultez l'article de blog Présentation des groupes de sécurité pour les Pods.

Considérations

- Avant de déployer des groupes de sécurité pour les Pods, tenez compte des limites et conditions suivantes :
- Les groupes de sécurité pour les Pods ne peuvent pas être utilisés avec les nœuds Windows.
- Les groupes de sécurité pour les Pods peuvent être utilisés avec des clusters configurés pour la famille `IPv6` qui contiennent des nœuds Amazon EC2 en utilisant la version 1.16.0 ou ultérieure du plug-in CNI Amazon VPC. Vous pouvez utiliser des groupes de sécurité pour les Pods avec des clusters configurés pour la famille `IPv6` qui contiennent des nœuds Fargate en utilisant la version 1.7.7 ou ultérieure du plug-in CNI Amazon VPC. Pour plus d'informations, consultez IPv6adresses pour les clustersPods, et services.
- Les groupes de sécurité pour Pods sont pris en charge par la plupart des familles d'instances Amazon EC2 basées sur Nitro, mais pas par toutes les générations d'une famille. Par exemple`m5`, la famille et les générations `c5` `r5`m6g,,c6g,, et d'`r6g`instance sont prises en charge. Aucun type d'instance de la famille `t` n'est pris en charge. Pour obtenir la liste complète des types d'instance pris en charge, veuillez consulter le fichier limits.go (français non disponible) sur GitHub. Vos nœuds doivent être l'un des types d'instance répertoriés qui ont `IsTrunkingCompatible: true` dans ce fichier.
- Si vous utilisez également des politiques de sécurité Pod pour restreindre l'accès à la mutation de Pod, l'utilisateur `eks:vpc-resource-controller` Kubernetes doit être spécifié dans le Kubernetes `ClusterRoleBinding` du `role` auquel votre `psp` est assigné. Si vous utilisez les objets par défaut `psp`, `role` et `ClusterRoleBinding` Amazon EKS, il s'agit de `eks:podsecuritypolicy:authenticated` `ClusterRoleBinding`. Par exemple, vous ajoutez l'utilisateur à la section `subjects:`, comme le montre l'exemple suivant :

```
[...]
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
  - apiGroup: rbac.authorization.k8s.io
    kind: User
    name: eks:vpc-resource-controller
  - kind: ServiceAccount
    name: eks-vpc-resource-controller
```

- Si vous utilisez les réseaux personnalisés et les groupes de sécurité pour Pods ensemble, le groupe de sécurité spécifié par les groupes de sécurité pour les Pods est utilisé à la place du groupe de sécurité spécifié dans le paramètre `ENIConfig`.
- Si vous utilisez la version `1.10.2` ou ultérieure du plugin CNI Amazon VPC et que vous incluez le paramètre `terminationGracePeriodSeconds` dans la spécification de votre Pod, la valeur du paramètre ne peut pas être égale à zéro.
- Si vous utilisez la version `1.10` ou antérieure du plugin CNI Amazon VPC, ou une version ultérieure `1.11` avec `POD_SECURITY_GROUP_ENFORCING_MODE=strict`, qui est le paramètre par défaut, alors les services Kubernetes de type `NodePort` et `LoadBalancer` utilisant des cibles d'instance avec un paramètre `externalTrafficPolicy` défini sur `Local` ne sont pas pris en charge avec les Pods que vous affectez aux groupes de sécurité. Pour plus d'informations sur l'utilisation d'un équilibreur de charge avec des cibles d'instance, consultez Répartition de charge réseau sur Amazon EKS
- Si vous utilisez la version `1.10` ou antérieure du plug-in Amazon VPC CNI ou la version `1.11` avec `POD_SECURITY_GROUP_ENFORCING_MODE=strict`, qui est le paramètre par défaut, le NAT source est désactivé pour le trafic sortant depuis les Pods avec des groupes de sécurité assignés afin que les règles sortantes de groupe de sécurité soient appliquées. Pour accéder à Internet, les Pods avec des groupes de sécurité assignés doivent être lancés sur des nœuds qui sont déployés dans un sous-réseau privé configuré avec une passerelle ou une instance NAT. Les Pods avec des groupes de sécurité assignés déployés sur des sous-réseaux publics ne peuvent pas accéder à Internet.

  Si vous utilisez la version `1.11` ou ultérieure du plug-in avec `POD_SECURITY_GROUP_ENFORCING_MODE=standard`, puis le trafic des Pod destiné à l'extérieur du VPC est traduit à l'adresse IP de l'interface réseau principale de l'instance. Pour ce trafic, les règles des groupes de sécurité de l'interface réseau principale sont utilisées, plutôt que les règles des groupes de sécurité de Pod's.
- Pour utiliser la stratégie réseau Calico avec les Pods qui ont des groupes de sécurité associés, vous devez utiliser la version `1.11.0` ou ultérieure du plugin CNI Amazon VPC et définir `POD_SECURITY_GROUP_ENFORCING_MODE=standard`. Sinon, le flux de trafic à destination et en provenance des Pods avec des groupes de sécurité associés n'est pas soumis à l'application de la stratégie réseau Calico et est limité à l'application du groupe de sécurité Amazon EC2 uniquement. Pour mettre à jour la version de votre plug-in CNI Amazon VPC, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS
- Les Pods exécutés sur des nœuds Amazon EC2 qui utilisent des groupes de sécurité dans des clusters qui utilisent Nodelocal DNSCache sont uniquement pris

en charge avec la version `1.11.0` ou ultérieure du plug-in Amazon VPC CNI et avec `POD_SECURITY_GROUP_ENFORCING_MODE=standard`. Pour mettre à jour la version de votre plug-in CNI Amazon VPC, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS

- Des groupes de sécurité pour les Pods peuvent entraîner une augmentation de la latence de démarrage des Pod pour des Pods avec un taux de désabonnement élevé. Cela est dû à la limitation du débit dans le contrôleur de ressources.

Configurer Amazon VPC CNI plugin for Kubernetes pour les groupes de sécurité des Pods

Pour déployer des groupes de sécurité pour les Pods

Si vous utilisez des groupes de sécurité pour les Pods Fargate uniquement et que vous n'avez aucun nœud Amazon EC2 dans votre cluster, passez à l'Déployer un exemple d'application.

1. Vérifiez la version actuelle de Amazon VPC CNI plugin for Kubernetes à l'aide de la commande suivante :

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
    | cut -d : -f 3
   ```

   L'exemple qui suit illustre un résultat.

   ```
   v1.7.6
   ```

   Si la version de votre Amazon VPC CNI plugin for Kubernetes est antérieure à `1.7.7`, mettez à jour le plugin vers la version `1.7.7` ou ultérieure. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

2. Ajoutez la politique IAM gérée `AmazonEKSVPCResourceController` au rôle de cluster qui est associé à votre cluster Amazon EKS. La politique permet au rôle de gérer les interfaces réseau, leurs adresses IP privées, leur attachement et leur détachement vers et depuis les instances réseau.

   a. Récupérez le nom de votre rôle IAM de cluster et stockez-le dans une variable. Remplacez *my-cluster* par le nom de votre cluster.

```
cluster_role=$(aws eks describe-cluster --name my-cluster --query
 cluster.roleArn --output text | cut -d / -f 2)
```

b. Attachez la stratégie au rôle.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSVPCResourceController --role-name $cluster_role
```

3. Activez le module complémentaire Amazon VPC CNI pour gérer les interfaces réseau pour les Pods en définissant la variable `ENABLE_POD_ENI` sur `true` dans le DaemonSet `aws-node`. Une fois que ce paramètre est défini sur `true`, le module complémentaire crée pour chaque nœud du cluster une ressource personnalisée cninode. Le contrôleur de ressources VPC crée et attache une interface réseau spéciale appelée interface réseau de tronc avec la description `aws-k8s-trunk-eni`.

```
kubectl set env daemonset aws-node -n kube-system ENABLE_POD_ENI=true
```

> **Note**
>
> L'interface réseau de tronc est incluse dans le nombre maximal d'interfaces réseau prises en charge par le type d'instance. Pour obtenir la liste du nombre maximal d'interfaces réseau prises en charge par chaque type d'instance, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de l'utilisateur Amazon EC2. Si votre nœud a déjà le nombre maximal d'interfaces réseau standard qui lui sont attachées, le contrôleur de ressources VPC réservera un espace. Vous devrez réduire suffisamment vos Pods en cours d'exécution pour que le contrôleur puisse détacher et supprimer une interface réseau standard, créer l'interface réseau de tronc et l'attacher à l'instance.

4. Vous pouvez voir quels nœuds ont une ressource personnalisée `CNINode` avec la commande suivante. Si le message `No resources found` est renvoyé, attendez plusieurs secondes et réessayez. L'étape précédente nécessite le redémarrage des Pods Amazon VPC CNI plugin for Kubernetes, qui prend plusieurs secondes.

```
$ kubectl get cninode -A
     NAME FEATURES
     ip-192-168-64-141.us-west-2.compute.internal
 [{"name":"SecurityGroupsForPods"}]
```

```
    ip-192-168-7-203.us-west-2.compute.internal [{"name":"SecurityGroupsForPods"}]
```

Si vous utilisez des versions CNI VPC antérieures à `1.15`, les étiquettes de nœud ont été utilisées à la place de la ressource personnalisée `CNINode`. Vous pouvez voir quels nœuds ont l'étiquette de nœud `aws-k8s-trunk-eni` définis sur `true` avec la commande suivante. Si le message `No resources found` est renvoyé, attendez plusieurs secondes et réessayez. L'étape précédente nécessite le redémarrage des Pods Amazon VPC CNI plugin for Kubernetes, qui prend plusieurs secondes.

```
kubectl get nodes -o wide -l vpc.amazonaws.com/has-trunk-attached=true
-
```

Une fois l'interface réseau de tronc créée, les Pods son assignés aux adresses IP secondaires à partir des interfaces réseau standard ou de connexion. L'interface de tronc est automatiquement supprimée si le nœud est supprimé.

Lorsque vous déployez un groupe de sécurité pour un Pod dans une étape ultérieure, le contrôleur de ressources VPC crée une interface réseau spéciale appelée interface réseau de branche avec une description de `aws-k8s-branch-eni` et y associe les groupes de sécurité. Les interfaces réseau de branche sont créées en plus des interfaces réseau standard et de tronc attachées au nœud.

Si vous utilisez des sondes de liveness ou de disponibilité, vous devez également désactiver le démux précoce TCP, de sorte que `kubelet` peut se connecter à des Pods sur des interfaces réseau de branche via TCP. Pour désactiver le démux précoce TCP, exécutez la commande suivante :

```
kubectl patch daemonset aws-node -n kube-system \
  -p '{"spec": {"template": {"spec": {"initContainers": [{"env":
[{"name":"DISABLE_TCP_EARLY_DEMUX","value":"true"}],"name":"aws-vpc-cni-
init"}]}}}}'
```

> **Note**
>
> Si vous utilisez la version `1.11.0` ou ultérieure du module complémentaire Amazon VPC CNI plugin for Kubernetes et définir sur

`POD_SECURITY_GROUP_ENFORCING_MODE=standard`, comme décrit à l'étape suivante, vous n'avez pas besoin d'exécuter la commande précédente.

5. Si votre cluster utilise `NodeLocal DNSCache`, ou si vous souhaitez utiliser la politique réseau Calico avec vos Pods qui ont leurs propres groupes de sécurité, ou si vous avez des services Kubernetes de type `NodePort` et `LoadBalancer` utilisant des cibles d'instance avec un paramètre `externalTrafficPolicy` défini sur `Local` pour les Pods que vous souhaitez attribuer à des groupes de sécurité, alors vous devez utiliser une version `1.11.0` ou ultérieure du module complémentaire Amazon VPC CNI plugin for Kubernetes et vous devez activer le paramètre suivant :

```
kubectl set env daemonset aws-node -n kube-system
 POD_SECURITY_GROUP_ENFORCING_MODE=standard
```

> **Important**
>
> - Les règles de groupe de sécurité de Pod ne sont pas appliquées au trafic entre les Pods ou entre les Pods et les services, comme `kubelet` ou `nodeLocalDNS`, qui se trouvent sur le même nœud. Les pods utilisant différents groupes de sécurité sur le même nœud ne peuvent pas communiquer, car ils sont configurés dans des sous-réseaux différents et le routage est désactivé entre ces sous-réseaux.
> - Le trafic sortant des Pods vers des adresses situées en dehors du VPC est l'adresse réseau traduite en adresse IP de l'interface réseau principale de l'instance (sauf si vous avez également défini `AWS_VPC_K8S_CNI_EXTERNALSNAT=true`). Pour ce trafic, les règles des groupes de sécurité de l'interface réseau principale sont utilisées, plutôt que les règles des groupes de sécurité de Pod's.
> - Pour que ce paramètre s'applique aux Pods existants, vous devez redémarrer les Pods ou les nœuds sur lesquels les Pods s'exécutent.

Déployer un exemple d'application

Pour utiliser des groupes de sécurité pour des Pods, vous devez avoir un groupe de sécurité existant et Déployer un Amazon EKS`SecurityGroupPolicy` sur votre cluster, comme décrit dans la procédure suivante. Les étapes suivantes vous montrent comment utiliser la politique de groupe de

sécurité pour un Pod. Sauf indication contraire, effectuez toutes les étapes à partir du même terminal, car les variables sont utilisées dans les étapes suivantes qui ne persistent pas entre les terminaux.

Pour déployer un exemple de Pod avec un groupe de sécurité

1. Créez un espace de noms Kubernetes vers lequel déployer les ressources. Vous pouvez remplacer *my-namespace* par le nom d'un espace de nom que vous voulez utiliser.

```
kubectl create namespace my-namespace
```

2. Déployez une politique `SecurityGroupPolicy` Amazon EKS sur votre cluster.

   a. Copiez les contenus suivants sur votre appareil. Vous pouvez remplacer *podSelector* par **serviceAccountSelector** si vous préférez sélectionner des Pods en fonction des labels de compte de service. Vous devez spécifier un sélecteur ou l'autre. Un `podSelector` vide (par exemple : `podSelector: {}`) sélectionne tous les Pods de l'espace de noms. Vous pouvez remplacer *my-role* par le nom de votre rôle. Un `serviceAccountSelector` vide sélectionne tous les comptes de service dans l'espace de noms. Vous pouvez remplacer *my-security-group-policy* par un nom pour votre `SecurityGroupPolicy` et *my-namespace* par le nom de l'espace de noms dans lequel vous voulez créer la `SecurityGroupPolicy`.

   Vous devez remplacer *my_pod_security_group_id* par l'ID d'un groupe de sécurité existant. Si vous n'avez pas de groupe de sécurité existant, vous devez en créer un. Pour plus d'informations, veuillez consulter la section Amazon EC2 security groups for Linux instances (français non garanti) dans le Guide de l'utilisateur Amazon EC2. Vous pouvez spécifier 1 à 5 ID de groupe de sécurité. Si vous spécifiez plusieurs ID, la combinaison de toutes les règles de tous les groupes de sécurité est effective pour les Pods sélectionnés.

```
cat >my-security-group-policy.yaml <<EOF
apiVersion: vpcresources.k8s.aws/v1beta1
kind: SecurityGroupPolicy
metadata:
  name: my-security-group-policy
  namespace: my-namespace
spec:
  podSelector:
    matchLabels:
      role: my-role
  securityGroups:
```

```
    groupIds:
      - my_pod_security_group_id
EOF
```

> **Important**
>
> Le ou les groupes de sécurité que vous spécifiez pour vos Pod doivent répondre aux critères suivants :
>
> - Ils doivent exister. S'ils n'existent pas, lorsque vous déployez un Pod correspondant au sélecteur, votre Pod reste bloqué dans le processus de création. Si vous décrivez le Pod, vous verrez un message d'erreur similaire à ce qui suit : `An error occurred (InvalidSecurityGroupID.NotFound) when calling the CreateNetworkInterface operation: The securityGroup ID 'sg-05b1d815d1EXAMPLE' does not exist.`
> - Ils doivent autoriser les communications entrantes du groupe de sécurité appliqué à vos nœuds (pour `kubelet`) sur tous les ports pour lesquels vous avez configuré des sondes.
> - Ils doivent autoriser les communications sortantes sur les ports `TCP` et `UDP` 53 vers un groupe de sécurité attribué aux Pods (ou aux nœuds sur lesquels les Pods sont exécutés) exécutant CoreDNS. Le groupe de sécurité pour vos Pods CoreDNS doit autoriser le trafic entrant des ports `TCP` et `UDP` 53 du groupe de sécurité que vous spécifiez.
> - Ils doivent disposer des règles entrantes et sortantes nécessaires pour communiquer avec d'autres Pods avec qui ils doivent communiquer.
> - Ils doivent avoir des règles qui permettent aux Pods de communiquer avec le plan de contrôle Kubernetes si vous utilisez le groupe de sécurité avec Fargate. La façon la plus simple de le faire est de spécifier le groupe de sécurité de cluster comme l'un des groupes de sécurité.
>
> Les politiques de groupe de sécurité s'appliquent uniquement aux Pods nouvellement planifiés. Elles n'affectent pas les Pods en cours d'exécution.

b. Déployez la politique.

```
kubectl apply -f my-security-group-policy.yaml
```

3. Déployez un exemple d'application avec une étiquette correspondant à la valeur *my-role* pour *podSelector* que vous avez spécifié à l'étape précédente.

   a. Copiez les contenus suivants sur votre appareil. Remplacez les *exemples de valeur* par les vôtres, puis exécutez la commande modifiée. Si vous remplacez *my-role*, assurez-vous qu'il est identique à la valeur que vous avez spécifiée pour le sélecteur dans une étape précédente.

```
cat >sample-application.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
  namespace: my-namespace
  labels:
    app: my-app
spec:
  replicas: 4
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        role: my-role
    spec:
      terminationGracePeriodSeconds: 120
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app
  namespace: my-namespace
  labels:
```

```
    app: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
EOF
```

b. Pour déployer l'application, exécutez la commande suivante. Lorsque vous déployez l'application, le plugin Amazon VPC CNI plugin for Kubernetes correspond au label `role` et les groupes de sécurité que vous avez spécifiés à l'étape précédente sont appliqués au Pod.

```
kubectl apply -f sample-application.yaml
```

4. Affichez les Pods déployés avec l'exemple d'application. Pour le reste de ce sujet, ce terminal est appelé `TerminalA`.

```
kubectl get pods -n my-namespace -o wide
```

L'exemple qui suit illustre un résultat.

```
NAME                             READY   STATUS    RESTARTS   AGE     IP
       NODE                                            NOMINATED NODE   READINESS
 GATES
my-deployment-5df6f7687b-4fbjm   1/1     Running   0          7m51s   192.168.53.48
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-j9fl4   1/1     Running   0          7m51s
  192.168.70.145   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-rjxcz   1/1     Running   0          7m51s
  192.168.73.207   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-zmb42   1/1     Running   0          7m51s   192.168.63.27
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
```

> **Note**
>
> - Si des Pods sont bloqués à l'état `Waiting`, exécutez `kubectl describe pod` ***`my-deployment-xxxxxxxxxx-xxxxx`*** `-n` *`my-namespace`*. Si vous voyez `Insufficient permissions: Unable to create Elastic Network Interface.`, vérifiez que vous avez ajouté la politique IAM au rôle de cluster IAM dans une étape précédente.
> - Si des Pods sont bloqués à l'état `Pending`, vérifiez que votre type d'instance de nœud est répertorié dans `limits.go` et que le produit du nombre maximal d'interfaces réseau de branche pris en charge par le type d'instance multiplié par le nombre de nœuds de votre groupe de nœuds n'a pas encore été atteint. Par exemple, une instance `m5.large` prend en charge neuf interfaces réseau de branche. Si votre groupe de nœuds comporte cinq nœuds, un maximum de 45 interfaces réseau de branche peut être créé pour le groupe de nœuds. Le 46e Pod que vous tentez de déployer sera installé dans l'état `Pending` jusqu'à ce qu'un autre Pod ayant des groupes de sécurité associés soit supprimé.

Si vous exécutez `kubectl describe pod` ***`my-deployment-xxxxxxxxxx-xxxxx`*** **`-n`** ***`my-namespace`*** et recevez un message similaire au message suivant, il peut être ignoré en toute sécurité. Ce message peut s'afficher lorsque Amazon VPC CNI plugin for Kubernetes tente de configurer les réseaux de l'hôte et échoue pendant la création de l'interface réseau. Le plugin journalise cet événement jusqu'à ce que l'interface réseau soit créée.

```
Failed to create Pod sandbox: rpc error: code = Unknown desc = failed to set up
 sandbox container
 "e24268322e55c8185721f52df6493684f6c2c3bf4fd59c9c121fd4cdc894579f" network for Pod
 "my-deployment-5df6f7687b-4fbjm": networkPlugin
cni failed to set up Pod "my-deployment-5df6f7687b-4fbjm-c89wx_my-namespace"
 network: add cmd: failed to assign an IP address to container
```

Vous ne pouvez pas dépasser le nombre maximal de Pods pouvant être exécutés sur le type d'instance. Pour obtenir la liste du nombre maximal de Pods que vous pouvez exécuter sur chaque type d'instance, consultez eni-max-pods.txt sur GitHub. Lorsque vous supprimez un Pod qui a des groupes de sécurité associés ou que vous supprimez le nœud sur lequel le Pod s'exécute, le contrôleur de ressources VPC supprime l'interface réseau de branche. Si

vous supprimez un cluster avec des Pods utilisant des Pods pour les groupes de sécurité, le contrôleur ne supprime pas les interfaces réseau de branche et vous devez donc les supprimer vous-même. Pour plus d'informations sur la suppression d'interfaces réseau, consultez Supprimer une interface réseau dans le guide de l'utilisateur Amazon EC2.

5. Dans un terminal séparé, placez-vous dans l'un des Pods. Pour le reste de ce sujet, ce terminal est appelé `TerminalB`. Remplacez *`5df6f7687b-4fbjm`* par l'ID d'un des Pods renvoyés dans votre sortie de l'étape précédente.

```
kubectl exec -it -n my-namespace my-deployment-5df6f7687b-4fbjm -- /bin/bash
```

6. À partir du shell dans `TerminalB`, confirmez que l'exemple d'application fonctionne.

```
curl my-app
```

L'exemple qui suit illustre un résultat.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

Vous avez reçu la sortie, car tous les Pods exécutant l'application sont associés au groupe de sécurité que vous avez créé. Ce groupe contient une règle qui autorise tout le trafic entre tous les Pods auxquels le groupe de sécurité est associé. Le trafic DNS est autorisé à partir de ce groupe de sécurité vers le groupe de sécurité du cluster associé à vos nœuds. Les nœuds exécutent les Pods CoreDNS, sur lesquels vos Pods ont effectué une recherche de nom.

7. À partir du `TerminalA`, supprimez les règles de groupe de sécurité qui autorisent la communication DNS au groupe de sécurité du cluster de votre groupe de sécurité. Si vous n'avez pas ajouté les règles DNS au groupe de sécurité du cluster lors d'une étape précédente, remplacez *`$my_cluster_security_group_id`* par l'ID du groupe de sécurité dans lequel vous avez créé les règles.

```
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_tcp_rule_id
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_udp_rule_id
```

8. À partir du `TerminalB`, essayez de nouveau d'accéder à l'application.

```
curl my-app
```

L'exemple qui suit illustre un résultat.

```
curl: (6) Could not resolve host: my-app
```

La tentative échoue, car le Pod n'est plus en mesure d'accéder aux Pods CoreDNS, auxquels le groupe de sécurité du cluster est associé. Le groupe de sécurité du cluster ne dispose plus des règles de groupe de sécurité qui autorisent la communication DNS à partir du groupe de sécurité associé à votre Pod.

Si vous tentez d'accéder à l'application à l'aide des adresses IP renvoyées pour l'un des Pods dans une étape précédente, vous recevez toujours une réponse, car tous les ports sont autorisés entre les Pods auxquels le groupe de sécurité est associé et une recherche de nom n'est pas requise.

9. Une fois l'expérimentation terminée, vous pouvez supprimer l'exemple de politique de groupe de sécurité, l'application et le groupe de sécurité que vous avez créés. Exécutez les commandes suivantes à partir du `TerminalA`.

```
kubectl delete namespace my-namespace
aws ec2 revoke-security-group-ingress --group-id $my_pod_security_group_id --
security-group-rule-ids $my_inbound_self_rule_id
wait
sleep 45s
aws ec2 delete-security-group --group-id $my_pod_security_group_id
```

Interfaces réseau multiples pour les Pods

Multus CNI est un plugin d'interface réseau de conteneur (Container Network Interface, CNI) pour Amazon EKS qui permet d'associer plusieurs interfaces réseau à un Pod. Pour plus d'informations, consultez la documentation Multus CNI sur GitHub.

Dans Amazon EKS, chaque Pod dispose d'une interface réseau attribuée par le plugin CNI Amazon VPC. Avec Multus, vous pouvez créer un Pod multi-homed qui possède plusieurs interfaces. Ceci est accompli par Multus agissant comme un « méta-plugin » ; un plugin CNI qui peut appeler plusieurs

autres plugins CNI. La prise en charge de AWS pour Multus est configurée avec le plugin CNI Amazon VPC comme plugin délégué par défaut.

Considérations

- Amazon EKS ne créera pas et ne publiera pas de plugins CNI pour la virtualisation d'I/O d'une racine unique (SR-IOV) et le kit de développement de données (DPDK). Toutefois, vous pouvez accélérer les paquets en vous connectant directement à Amazon EC2 Elastic Network Adapters (ENA) via le périphérique hôte géré par Multus et les plugins `ipvlan`.
- Amazon EKS prend en charge Multus, qui fournit un processus générique qui permet un chaînage simple de plugins CNI supplémentaires. Multus et le processus de chaînage est pris en charge, mais AWS ne prendra pas en charge tous les plugins CNI compatibles qui peuvent être chaînés, ou les problèmes qui peuvent survenir dans ces plugins CNI qui ne sont pas liés à la configuration de chaînage.
- Amazon EKS assure la prise en charge et la gestion du cycle de vie du plugin Multus, mais n'est pas responsable des adresses IP ou de la gestion supplémentaire associée aux interfaces réseau supplémentaires. L'adresse IP et la gestion de l'interface réseau par défaut utilisant le plugin CNI Amazon VPC restent inchangées.
- Seul le plugin CNI Amazon VPC est officiellement pris en charge en tant que plugin délégué par défaut. Vous devez modifier le manifeste d'installation de Multus publié pour reconfigurer le plugin délégué par défaut vers un autre CNI si vous choisissez de ne pas utiliser le plugin CNI Amazon VPC pour la mise en réseau principale.
- Multus n'est pris en charge que lorsque vous utilisez le CNI d'Amazon VPC comme CNI principal. Nous ne prenons pas en charge le CNI d'Amazon VPC lorsqu'il est utilisée pour des interfaces d'ordre supérieur, secondaires ou autres.
- Pour empêcher le plugin CNI Amazon VPC d'essayer de gérer des interfaces réseau supplémentaires attribuées aux Pods, ajoutez la balise suivante à l'interface réseau :

  Clé : `node.k8s.amazonaws.com/no_manage`

  valeur : `true`
- Multus est compatible avec les politiques réseau, mais la politique doit être enrichie pour inclure les ports et les adresses IP qui peuvent faire partie d'interfaces réseau supplémentaires attachées aux Pods.

Pour une démonstration de la mise en œuvre, consultez le Guide d'installation de Multus sur GitHub.

Autres plugins CNI compatibles

Le seul plugin CNI pris en charge par Amazon EKS est Amazon VPC CNI plugin for Kubernetes. Amazon EKS fonctionne en amont de Kubernetes, vous pouvez donc installer d'autres plugins CNI compatibles sur les nœuds Amazon EC2 de votre cluster. Si vous avez des nœuds Fargate dans votre cluster, alors Amazon VPC CNI plugin for Kubernetes se trouve déjà sur vos nœuds Fargate. C'est le seul plugin CNI que vous pouvez utiliser avec les nœuds Fargate. Une tentative d'installation d'un autre plugin CNI sur les nœuds Fargate a échoué.

Si vous prévoyez d'utiliser un autre plugin CNI sur des nœuds Amazon EC2, nous vous recommandons d'obtenir un support commercial pour le plugin ou de disposer de l'expertise interne pour dépanner et contribuer aux corrections du projet de plugin CNI.

Amazon EKS entretient des relations avec un réseau de partenaires qui offrent un support pour d'autres plugins CNI compatibles. Pour plus d'informations sur les versions, les qualifications et les tests effectués, consultez la documentation des partenaires suivante.

Partenaire	Produit (langue française non garantie)	Documentation
Tigera	Calico	Instructions d'installation
Isovalent	Cilium	Instructions d'installation
Juniper	Réseau Contrail natif cloud (CN2)	Instructions d'installation
VMware	Antrea	Instructions d'installation

Amazon EKS vise à vous offrir un large choix d'options pour couvrir tous les cas d'utilisation.

Autres plugins de politique réseau compatibles

Calico est une solution largement adoptée pour la mise en réseau et la sécurité des conteneurs. L'utilisation Calico sur EKS permet d'appliquer une politique réseau totalement conforme à vos clusters EKS. En outre, vous pouvez choisir d'utiliser Calico le réseau, qui conserve les adresses IP de votre VPC sous-jacent. Calico Cloud améliore les fonctionnalités deCalico Open Source, en fournissant des fonctionnalités avancées de sécurité et d'observabilité.

Qu'est-ce que AWS Load Balancer Controller ?

AWS Load Balancer Controllergère les équilibreurs de charge AWS élastiques pour un Kubernetes cluster. Vous pouvez utiliser le contrôleur pour exposer les applications de votre cluster à Internet. Le contrôleur fournit des équilibreurs de AWS charge qui pointent vers les ressources de service ou d'entrée du cluster. En d'autres termes, le contrôleur crée une adresse IP ou un nom DNS unique qui pointe vers plusieurs pods de votre cluster.

Le contrôleur surveille Kubernetes Ingress nos Service ressources. En réponse, il crée les ressources AWS Elastic Load Balancing appropriées. Vous pouvez configurer le comportement spécifique des équilibreurs de charge en appliquant des annotations aux Kubernetes ressources. Par exemple, vous pouvez associer des groupes AWS de sécurité aux équilibreurs de charge à l'aide d'annotations.

Le contrôleur fournit les ressources suivantes :

Kubernetes `Ingress`

Le LBC crée un AWS Application Load Balancer (ALB) lorsque vous créez un. Kubernetes `Ingress` Passez en revue les annotations que vous pouvez appliquer à une ressource Ingress.

Service Kubernetes de type `LoadBalancer`

Le LBC crée un AWS Network Load Balancer (NLB) lorsque vous créez Kubernetes un service de type. `LoadBalancer` Passez en revue les annotations que vous pouvez appliquer à une ressource de service.

Dans le passé, l'équilibreur de charge Kubernetes réseau était utilisé pour les cibles d'instance, mais le LBC était utilisé pour les cibles IP. Avec AWS Load Balancer Controller, version `2.3.0` ou ultérieure, vous pouvez créer des dispositifs d'équilibrage de charge de réseau à l'aide de l'un ou l'autre type de cible. Pour plus d'informations sur les types de cibles NLB, consultez Type de cible dans le guide de l'utilisateur de Network Load Balancer.

Le contrôleur est un projet open source géré surGitHub.

Avant de déployer le contrôleur, nous vous recommandons de consulter les prérequis et les considérations dans Répartition de la charge des applications sur Amazon EKS et Répartition de charge réseau sur Amazon EKS. Dans ces rubriques, vous allez déployer un exemple d'application qui inclut un équilibreur de AWS charge.

Installation du contrôleur

- Apprenez commentthe section called "Installation avec Helm".Utilisez cette procédure si vous utilisez Amazon EKS pour la première fois. Cette procédure utilise Helm, un gestionnaire de packages pour Kubernetes et `eksctl`pour simplifier l'installation du LBC.
- Sinon,the section called "Installation à l'aide de manifestes". Cette procédure convient aux configurations de cluster avancées. Cela inclut les clusters avec un accès réseau restreint aux registres de conteneurs publics.

Migrer depuis des versions de contrôleur obsolètes

- Si vous disposez de versions obsolètes du système AWS Load Balancer Controller installé, découvrez comment procéder. the section called "Migrer depuis un contrôleur obsolète"
- Les versions obsolètes ne peuvent pas être mises à niveau. Ils doivent être supprimés et une version à jour doit être AWS Load Balancer Controller installée.
- Les versions obsolètes incluent :
  - AWS Contrôleur d'entrée ALB pour Kubernetes (« Ingress Controller »), prédécesseur du. AWS Load Balancer Controller
  - N'importe quelle `0.1.`*x* version du AWS Load Balancer Controller

Fournisseur de cloud traditionnel

Kubernetesinclut un ancien fournisseur de cloud pour AWS. L'ancien fournisseur de cloud est capable de provisionner des équilibreurs de AWS charge, similaires au. AWS Load Balancer Controller L'ancien fournisseur de cloud crée des équilibreurs de charge classiques. Si vous n'installez pas leAWS Load Balancer Controller, vous Kubernetes utiliserez par défaut l'ancien fournisseur de cloud. Vous devez installer l'ancien fournisseur de cloud AWS Load Balancer Controller et éviter de l'utiliser.

> **Important**
>
> Dans les versions 2.5 et ultérieures, AWS Load Balancer Controller il devient le contrôleur par défaut pour les ressources de Kubernetes service avec le `type: LoadBalancer` et crée un AWS Network Load Balancer (NLB) pour chaque service. Pour ce faire, il génère un webhook de mutation pour les services, qui définit le champ `spec.loadBalancerClass` sur `service.k8s.aws/nlb` pour les nouveaux services d'équilibreur `type:`

`LoadBalancer`. Vous pouvez désactiver cette fonctionnalité et revenir à l'utilisation de l'ancien fournisseur de cloud en tant que contrôleur par défaut, en définissant la valeur `enableServiceMutatorWebhook` des charts de Helm sur `false`. À moins que vous ne désactiviez cette fonctionnalité, le cluster ne générera pas de nouveaux équilibreurs de charge Classic Load Balancer pour vos services. Les équilibreurs de charge Classic Load Balancer existants continueront de fonctionner.

Installez le Helm AWS Load Balancer Controller à l'aide

Cette rubrique décrit comment installer Helm à l'AWS Load Balancer Controlleraide de Helm, un gestionnaire de packages pourKubernetes, et`eksctl`. Le contrôleur est installé avec les options par défaut. Pour plus d'informations sur le contrôleur, y compris des détails sur sa configuration avec des annotations, consultez la AWS Load Balancer Controllerdocumentation surGitHub.

Dans les étapes suivantes, remplacez *`example values`* par vos propres valeurs.

Prérequis

Avant de démarrer ce didacticiel, vous devez installer et configurer les outils et les ressources suivants dont vous avez besoin pour créer et gérer un cluster Amazon EKS.

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.
- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Assurez-vous que les versions de vos modules complémentaires Amazon VPC CNI plugin for Kubernetes, `kube-proxy`, et CoreDNS correspondent au moins à celles répertoriées dans Jetons de compte de service.
- Connaissance d' AWS Elastic Load Balancing. Pour plus d'informations, consultez le Guide de l'utilisateur Elastic Load Balancing.
- Connaissance du service Kubernetes et des ressources ingress.

- Helm installé localement.

Étape 1 : créer un rôle IAM à l'aide de **eksctl**

> **Note**
>
> Il vous suffit de créer un rôle IAM pour AWS Load Balancer Controller un rôle par AWS compte. Vérifiez s'`AmazonEKSLoadBalancerControllerRole`il existe dans la console IAM. Si ce rôle existe, passez directement àthe section called "Étape 2 : Installation AWS Load Balancer Controller".

Créez une politique IAM.

1. Téléchargez une politique IAM pour le AWS Load Balancer Controller afin de lui permettre d'effectuer des appels aux API AWS en votre nom.

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. Créez une politique IAM à l'aide de la politique téléchargée à l'étape précédente.

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

   > **Note**
   >
   > Si vous consultez la politique dans le AWS Management Console, la console affiche des avertissements pour le service ELB, mais pas pour le service ELB v2. Cela est dû au fait

que certaines des actions de la politique sont disponibles pour ELB v2, mais pas pour ELB. Vous pouvez ignorer les avertissements relatifs au service ELB.

Créez un rôle IAM à l'aide de **`eksctl`**

- Remplacez *`my-cluster`* par le nom de votre cluster, *`111122223333`* par votre ID de compte, puis exécutez la commande. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
$ eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

Étape 2 : Installation AWS Load Balancer Controller

Installation à AWS Load Balancer Controller l'aide de Helm V3

1. Ajoutez le référentiel de `eks-charts` diagrammes Helm. AWS maintient ce dépôt activé GitHub.

```
$ helm repo add eks https://aws.github.io/eks-charts
```

2. Mettez à jour votre référentiel local pour vous assurer que vous disposez des graphiques les plus récents.

```
$ helm repo update eks
```

3. Installez la AWS Load Balancer Controller.

   Remplacez *`my-cluster`* par le nom de votre cluster. Dans la commande suivante, `aws-load-balancer-controller` est le compte de service Kubernetes que vous avez créé à l'étape précédente.

   Pour plus d'informations sur la configuration de l'organigramme, reportez-vous à `values.yaml`la section suivante GitHub.

```
$ helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=my-cluster \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

a. Si vous déployez le contrôleur sur des nœuds Amazon EC2 qui ont un accès limité au service de métadonnées d'instance Amazon EC2 (IMDS), ou si vous déployez sur Fargate, ajoutez les indicateurs suivants à la commande `helm` suivante :

   - **`--set region=region-code`**
   - **`--set vpcId=vpc-xxxxxxxx`**

b. Pour consulter les versions disponibles du Helm Chart et du Load Balancer Controller, utilisez la commande suivante :

```
helm search repo eks/aws-load-balancer-controller --versions
```

> **Important**
>
> Le graphique déployé ne reçoit pas automatiquement les mises à jour de sécurité. Vous devez effectuer manuellement une mise à niveau vers un graphique plus récent lorsqu'il sera disponible. Lors de la mise *`install`* à niveau, **`upgrade`** passez à la commande précédente.
>
> La `helm install` commande installe automatiquement les définitions de ressources personnalisées (CRDs) pour le contrôleur. Ce n'est pas le cas de la `helm upgrade` commande. Si vous utilisez, `helm upgrade,` vous devez installer manuellement leCRDs. Exécutez la commande suivante pour installer CRDs :
>
> ```
> wget https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml
> kubectl apply -f crds.yaml
> ```

Étape 3 : vérifier que le contrôleur est installé

1. Vérifiez que le contrôleur est installé.

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

L'exemple qui suit illustre un résultat.

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

Vous recevez la sortie précédente si vous avez déployé à l'aide de Helm. Si vous avez déployé à l'aide du manifeste Kubernetes, vous n'avez qu'un seul réplica.

2. Avant d'utiliser le contrôleur pour provisionner AWS des ressources, votre cluster doit répondre à des exigences spécifiques. Pour plus d'informations, consultez Répartition de la charge des applications sur Amazon EKS et Répartition de charge réseau sur Amazon EKS.

Installez le AWS Load Balancer Controller module complémentaire à l'aide de Kubernetes Manifests

Cette rubrique décrit comment installer le contrôleur en téléchargeant et en appliquant des Kubernetes manifestes. Vous pouvez afficher l'intégralité de la documentation pour le contrôleur sur GitHub.

Dans les étapes suivantes, remplacez *example values* par vos propres valeurs.

Prérequis

Avant de démarrer ce didacticiel, vous devez installer et configurer les outils et les ressources suivants dont vous avez besoin pour créer et gérer un cluster Amazon EKS.

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.
- Un fournisseur AWS Identity and Access Management (IAM) OpenID Connect (OIDC) existant pour votre cluster. Pour déterminer si vous en avez déjà un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
- Assurez-vous que les versions de vos modules complémentaires Amazon VPC CNI plugin for Kubernetes, `kube-proxy`, et CoreDNS correspondent au moins à celles répertoriées dans Jetons de compte de service.
- Connaissance d' AWS Elastic Load Balancing Pour plus d'informations, consultez le Guide de l'utilisateur Elastic Load Balancing.

- Connaissance du service Kubernetes et des ressources ingress.

Étape 1 : Configuration de l'IAM

> **Note**
>
> Il vous suffit de créer un rôle IAM pour AWS Load Balancer Controller un rôle par AWS compte. Vérifiez s'`AmazonEKSLoadBalancerControllerRole`il existe dans la console IAM. Si ce rôle existe, passez directement àthe section called “Étape 2 : Installation `cert-manager`”.

Créez une politique IAM.

1. Téléchargez une politique IAM pour le AWS Load Balancer Controller afin de lui permettre d'effectuer des appels aux API AWS en votre nom.

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. Créez une politique IAM à l'aide de la politique téléchargée à l'étape précédente.

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

> **Note**
>
> Si vous consultez la politique dans le AWS Management Console, la console affiche des avertissements pour le service ELB, mais pas pour le service ELB v2. Cela est dû au fait que certaines des actions de la politique sont disponibles pour ELB v2, mais pas pour ELB. Vous pouvez ignorer les avertissements relatifs au service ELB.

eksctl

Créez un rôle IAM à l'aide de **eksctl**

- Remplacez *my-cluster* par le nom de votre cluster, *111122223333* par votre ID de compte, puis exécutez la commande. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
$ eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

AWS CLI and kubectl

Créez un rôle IAM à l'aide du et AWS CLI**kubectl**

1. Récupérez l'ID de fournisseur OIDC de votre cluster et stockez-le dans une variable.

```
oidc_id=$(aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

2. Déterminez si un fournisseur OIDC IAM avec l'identifiant de votre cluster est déjà présent sur votre compte. Vous devez être OIDC configuré à la fois pour le cluster et pour IAM.

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Si une sortie est renvoyée, cela signifie que vous disposez déjà d'un fournisseur OIDC IAM pour votre cluster. Si aucune sortie n'est renvoyée, vous devez créer un fournisseur OIDC IAM pour votre cluster. Pour plus d'informations, consultez Créez un OIDC fournisseur IAM pour votre cluster.

3. Copiez les contenus suivants sur votre appareil. Remplacez *111122223333* par votre ID de compte. *region-code*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster. Remplacez *EXAMPLED539D4633E53DE1B71EXAMPLE* par la sortie renvoyée dans l'étape précédente. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:` Après avoir remplacé le texte, exécutez la commande modifiée pour créer le fichier `load-balancer-role-trust-policy.json`.

```
cat >load-balancer-role-trust-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-load-balancer-controller"
                }
            }
        }
    ]
}
EOF
```

4. Créez le rôle IAM.

```
aws iam create-role \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --assume-role-policy-document file://"load-balancer-role-trust-policy.json"
```

5. Attachez la politique IAM gérée par Amazon EKS au rôle IAM. Remplacez *111122223333* par votre ID de compte.

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --role-name AmazonEKSLoadBalancerControllerRole
```

6. Copiez les contenus suivants sur votre appareil. Remplacez *111122223333* par votre ID de compte. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:` Après avoir remplacé le texte, exécutez la commande modifiée pour créer le fichier `aws-load-balancer-controller-service-account.yaml`.

```
cat >aws-load-balancer-controller-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/AmazonEKSLoadBalancerControllerRole
EOF
```

7. Créez le compte de service Kubernetes sur votre cluster. Le compte de service Kubernetes nommé `aws-load-balancer-controller` est annoté avec le rôle IAM que vous avez créé nommé *AmazonEKSLoadBalancerControllerRole*.

```
$ kubectl apply -f aws-load-balancer-controller-service-account.yaml
```

Étape 2 : Installation **`cert-manager`**

Installez `cert-manager` en utilisant l'une des méthodes suivantes pour injecter la configuration du certificat dans les webhooks. Pour plus d'informations, consultez le guide de démarrage de la documentation du *`cert-manager`*.

Nous vous recommandons d'utiliser le registre des `quay.io` conteneurs pour l'installation`cert-manager`. Si vos nœuds n'ont pas accès au registre des `quay.io` conteneurs, installez-le à `cert-manager` l'aide d'Amazon ECR (voir ci-dessous).

Quay .io

Installation à l'**`cert-manager`**aide de Quay.io

- Si vos nœuds ont accès au registre de conteneurs `quay.io`, installez `cert-manager` pour injecter la configuration du certificat dans les webhooks.

```
$ kubectl apply \
    --validate=false \
    -f https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-manager.yaml
```

Amazon ECR

Installation à **`cert-manager`** l'aide d'Amazon ECR

1. Installez `cert-manager` en utilisant l'une des méthodes suivantes pour injecter la configuration du certificat dans les webhooks. Pour plus d'informations, consultez le guide de démarrage de la documentation du *`cert-manager`*.
2. Téléchargez le manifeste.

```
curl -Lo cert-manager.yaml https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-manager.yaml
```

3. Extrayez les images suivantes et transférez-les vers un référentiel auquel vos nœuds ont accès. Pour plus d'informations sur l'extraction, le balisage et le transfert d'images vers votre propre référentiel, consultez Copier une image de conteneur d'un référentiel vers un autre référentiel.

```
quay.io/jetstack/cert-manager-cainjector:v1.13.5
quay.io/jetstack/cert-manager-controller:v1.13.5
quay.io/jetstack/cert-manager-webhook:v1.13.5
```

4. Remplacez `quay.io` dans le manifeste des trois images par votre propre nom de registre. La commande suivante suppose que le nom de votre référentiel privé est le même que celui du référentiel source. Remplacez *111122223333.dkr.ecr.region-code.amazonaws.com* par votre registre privé.

```
$ sed -i.bak -e 's|quay.io|111122223333.dkr.ecr.region-code.amazonaws.com|' ./
cert-manager.yaml
```

5. Appliquez le manifeste.

```
$ kubectl apply \
    --validate=false \
    -f ./cert-manager.yaml
```

Étape 3 : Installation AWS Load Balancer Controller

Installation AWS Load Balancer Controller à l'aide d'un Kubernetes manifeste

1. Téléchargez la spécification du contrôleur. Pour plus d'informations sur le contrôleur, consultez la documentation sur GitHub.

```
curl -Lo v2_7_2_full.yaml https://github.com/kubernetes-sigs/aws-load-balancer-
controller/releases/download/v2.7.2/v2_7_2_full.yaml
```

2. Effectuez les modifications suivantes dans le fichier.

   a. Si vous avez téléchargé le fichier `v2_7_2_full.yaml`, exécutez la commande suivante pour supprimer la section `ServiceAccount` du manifeste. Si vous ne supprimez pas cette section, l'annotation requise que vous avez apportée au compte de service lors d'une étape précédente sera écrasée. La suppression de cette section préserve également le compte de service que vous avez créé à une étape précédente, si vous supprimez le contrôleur.

```
$ sed -i.bak -e '596,604d' ./v2_7_2_full.yaml
```

Si vous avez téléchargé une version différente du fichier, ouvrez le fichier dans un éditeur et supprimez les lignes suivantes.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
---
```

b. Remplacez `your-cluster-name` dans la section `Deployment spec` du fichier par le nom de votre cluster en remplaçant *`my-cluster`* par le nom de votre cluster.

```
$ sed -i.bak -e 's|your-cluster-name|my-cluster|' ./v2_7_2_full.yaml
```

c. Si vos nœuds n'ont pas accès aux référentiels d'images Amazon EKS Amazon ECR, vous devez extraire l'image suivante et l'envoyer vers un référentiel auquel vos nœuds ont accès. Pour plus d'informations sur l'extraction, l'identification et le transfert d'une image vers votre propre référentiel, consultez Copier une image de conteneur d'un référentiel vers un autre référentiel.

```
public.ecr.aws/eks/aws-load-balancer-controller:v2.7.2
```

Ajoutez le nom de votre registre au manifeste. La commande suivante suppose que le nom de votre référentiel privé est le même que celui du référentiel source et ajoute le nom de votre registre privé au fichier. Remplacez *`111122223333.dkr.ecr.region-code.amazonaws.com`* par votre registre. Cette ligne suppose que vous avez nommé votre référentiel privé de la même manière que le référentiel source. Dans le cas contraire, remplacez le texte `eks/aws-load-balancer-controller` après votre nom de registre privé par le nom de votre référentiel.

```
$ sed -i.bak -e 's|public.ecr.aws/eks/aws-load-balancer-
controller|111122223333.dkr.ecr.region-code.amazonaws.com/eks/aws-load-
balancer-controller|' ./v2_7_2_full.yaml
```

d. (Requis uniquement pour Fargate ou Restricted IMDS)

Si vous déployez le contrôleur sur les nœuds Amazon EC2 qui ont un accès limité au service des métadonnées d'instance Amazon EC2 (IMDS), ou si vous déployez sur Fargate, ajoutez l'option **`following parameters`** dans `- args:`.

```
[...]
spec:
      containers:
        - args:
            - --cluster-name=your-cluster-name
            - --ingress-class=alb
            - --aws-vpc-id=vpc-xxxxxxxx
            - --aws-region=region-code


[...]
```

3. Appliquez le fichier.

```
$ kubectl apply -f v2_7_2_full.yaml
```

4. Téléchargez l'attaque de l'homme du milieu (HDM) `IngressClass` et `IngressClassParams` à votre cluster.

```
$ curl -Lo v2_7_2_ingclass.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/download/v2.7.2/v2_7_2_ingclass.yaml
```

5. Appliquez le manifeste à votre cluster.

```
$ kubectl apply -f v2_7_2_ingclass.yaml
```

Étape 4 : vérifier que le contrôleur est installé

1. Vérifiez que le contrôleur est installé.

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

L'exemple qui suit illustre un résultat.

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
```

```
aws-load-balancer-controller   2/2     2            2           84s
```

Vous recevez la sortie précédente si vous avez déployé à l'aide de Helm. Si vous avez déployé à l'aide du manifeste Kubernetes, vous n'avez qu'un seul réplica.

2. Avant d'utiliser le contrôleur pour provisionner AWS des ressources, votre cluster doit répondre à des exigences spécifiques. Pour plus d'informations, consultez Répartition de la charge des applications sur Amazon EKS et Répartition de charge réseau sur Amazon EKS.

Migrer depuis un contrôleur obsolète

Cette rubrique décrit comment effectuer une migration à partir de versions de contrôleurs obsolètes. Plus précisément, il décrit comment supprimer les versions obsolètes du. AWS Load Balancer Controller

- Les versions obsolètes ne peuvent pas être mises à niveau. Ils doivent être supprimés et une version actuelle du LBC doit être installée.
- Les versions déconseillées incluent :
  - AWS Contrôleur d'entrée ALB pour Kubernetes (« Ingress Controller »), prédécesseur du. AWS Load Balancer Controller
  - N'importe quelle `0.1.`*x* version du AWS Load Balancer Controller

Supprimer la version obsolète du contrôleur

> **Note**
>
> Vous avez peut-être installé la version obsolète à l'aide de Helm ou manuellement à l'aide de manifestes. Kubernetes Terminez la procédure à l'aide de l'outil avec lequel vous l'avez installé à l'origine.

Supprimer le contrôleur d'entrée à l'aide de Helm

1. Si vous avez installé les Charts de Helm `incubator/aws-alb-ingress-controller`, désinstallez-les.

```
$ helm delete aws-alb-ingress-controller -n kube-system
```

2. Si la version `0.1.x` du graphique `eks-charts/aws-load-balancer-controller` est installée, désinstallez-la. La mise à niveau depuis `0.1.x` vers la version `1.0.0` ne fonctionne pas en raison d'une incompatibilité avec la version de l'API du webhook.

```
$ helm delete aws-load-balancer-controller -n kube-system
```

Supprimer le contrôleur d'entrée à l'aide d'un manifeste Kubernetes

1. Vérifiez si le contrôleur est actuellement installé.

```
$ kubectl get deployment -n kube-system alb-ingress-controller
```

   Il s'agit de la sortie si le contrôleur n'est pas installé.

   Erreur provenant du serveur (NotFound) : deployments.apps « » alb-ingress-controller est introuvable

   Il s'agit de la sortie si le contrôleur est installé.

```
NAME                   READY UP-TO-DATE AVAILABLE AGE
alb-ingress-controller 1/1   1          1         122d
```

2. Saisissez les commandes suivantes pour supprimer le contrôleur.

```
$ kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
ingress-controller/v1.1.8/docs/examples/alb-ingress-controller.yaml
kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
ingress-controller/v1.1.8/docs/examples/rbac-role.yaml
```

Migrer vers AWS Load Balancer Controller

Pour migrer de l'ALB Ingress Controller Kubernetes vers leAWS Load Balancer Controller, vous devez :

1. Retirez le contrôleur d'entrée ALB (voir ci-dessus).
2. Installez leAWS Load Balancer Controller.
3. Ajoutez une politique supplémentaire au rôle IAM utilisé par le LBC. Cette politique permet au LBC de gérer les ressources créées par le contrôleur d'entrée ALB pour. Kubernetes

Ajoutez une politique de migration au rôle AWS Load Balancer Controller IAM.

1. Téléchargez la politique IAM. Cette politique permet au LBC de gérer les ressources créées par le contrôleur d'entrée ALB pour. Kubernetes Vous pouvez également afficher la politique.

```
$ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/v2.7.2/docs/install/iam_policy_v1_to_v2_additional.json
```

2. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:` .

```
$ sed -i.bak -e 's|arn:aws:|arn:aws-us-gov:|' iam_policy_v1_to_v2_additional.json
```

3. Créez la politique IAM et notez l'ARN renvoyé.

```
$ aws iam create-policy \
  --policy-name AWSLoadBalancerControllerAdditionalIAMPolicy \
  --policy-document file://iam_policy_v1_to_v2_additional.json
```

4. Attachez la politique IAM au rôle IAM utilisé par le LBC. Remplacez *`your-role-name`* par le nom du rôle, tel que`AmazonEKSLoadBalancerControllerRole`.

   Si vous avez créé le rôle en utilisant`eksctl`, pour trouver le nom du rôle créé, ouvrez la AWS CloudFormation console et sélectionnez la pile eksctl- ***`my-cluster`*** - - addon-iamserviceaccount-kube-system. aws-load-balancer-controller Sélectionnez l'onglet Resources (Ressources). Le nom du rôle se trouve dans la colonne ID Physique. Si votre cluster se trouve dans AWS GovCloud (USA Est) ou AWS GovCloud (USA Ouest) Régions AWS, remplacez-le par`arn:aws:`. `arn:aws-us-gov:`

```
$ aws iam attach-role-policy \
  --role-name your-role-name \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerAdditionalIAMPolicy
```

Utilisation du module complémentaire CoreDNS Amazon EKS

CoreDNS est un serveur DNS flexible et extensible qui peut servir de DNS de cluster Kubernetes. Lorsque vous lancez un cluster Amazon EKS avec au moins un nœud, deux réplicas de l'image CoreDNS sont déployés par défaut, quel que soit le nombre de nœuds déployés dans votre cluster.

Les Pods CoreDNS fournissent une résolution de noms pour tous les Pods du cluster. Les Pods CoreDNS peuvent être déployés sur les nœuds Fargate si votre cluster inclut un AWS Fargate profil avec un espace de noms qui correspond à l'espace de noms pour le CoreDNS `deployment`. Pour plus d'informations sur CoreDNS, consultez Utilisation de CoreDNS pour la découverte des services dans la documentation Kubernetes.

Le tableau suivant répertorie la dernière version du module complémentaire Amazon EKS pour chaque version de Kubernetes.

Version de Kubernetes	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
	v1.11.1-eksbuild	v1.11.1-eksbuild	v1.10.1-eksbuild.1	v1.10.1-eksbuild.1	v1.9.3-eksbuild.	v1.9.3-eksbuild.	v1.9.3-eksbuild.	v1.8.7-eksbuild.10

> **Important**
>
> Si vous gérez vous-même ce module complémentaire, les versions répertoriées dans le tableau peuvent ne pas être les mêmes que les versions autogérées disponibles. Pour plus d'informations sur la mise à jour du type autogéré de ce module complémentaire, consultez la rubrique Mise à jour du module complémentaire autogéré.

Considérations importantes concernant la mise à niveau de CoreDNS

- Pour améliorer la stabilité et la disponibilité de CoreDNSDeployment, des versions `v1.9.3-eksbuild.6` et des versions ultérieures et, `v1.10.1-eksbuild.3` elles sont déployées avec un `PodDisruptionBudget`. Si vous avez déployé une version existante `PodDisruptionBudget`, la mise à niveau vers ces versions risque d'échouer. Si la mise à niveau échoue, l'exécution de l'une des tâches suivantes devrait résoudre le problème :
  - Lors de la mise à niveau du module complémentaire Amazon EKS, choisissez de remplacer les paramètres existants comme option de résolution des conflits. Si vous avez défini d'autres paramètres personnalisés pour le Deployment, assurez-vous de les sauvegarder avant de

procéder à la mise à niveau afin de pouvoir réappliquer vos autres paramètres personnalisés après la mise à niveau.

- Supprimez votre version existante `PodDisruptionBudget` et réessayez d'effectuer la mise à niveau.

- Dans les versions `v1.9.3-eksbuild.3` et ultérieures, et `v1.10.1-eksbuild.6` et ultérieures des modules complémentaires EKS, le Deployment CoreDNS définit `readinessProbe` pour utiliser le point de terminaison `/ready`. Ce point de terminaison est activé dans le fichier de configuration `Corefile` pour CoreDNS.

  Si vous utilisez un `Corefile` personnalisé, vous devez ajouter le plug-in `ready` à la configuration, afin que le point de terminaison `/ready` soit actif dans CoreDNS pour que la sonde puisse l'utiliser.

- Dans les versions `v1.9.3-eksbuild.7` et ultérieures, et `v1.10.1-eksbuild.4` et ultérieures des modules complémentaires EKS, vous pouvez modifier le `PodDisruptionBudget`. Vous pouvez modifier le module complémentaire et modifier ces paramètres dans les paramètres de configuration facultatifs à l'aide des champs de l'exemple suivant. Cet exemple montre la valeur `PodDisruptionBudget` par défaut.

  ```
  {
      "podDisruptionBudget": {
          "enabled": true,
          "maxUnavailable": 1
          }
  }
  ```

  Vous pouvez définir `maxUnavailable` ou `minAvailable`, mais vous ne pouvez pas définir les deux dans un même `PodDisruptionBudget`. Pour plus d'informations sur `PodDisruptionBudgets`, consultez la rubrique Spécification d'un `PodDisruptionBudget` de la documentation Kubernetes.

  Notez que si vous définissez `enabled` sur `false`, le `PodDisruptionBudget` n'est pas supprimé. Après avoir défini ce champ sur `false`, vous devez supprimer l'objet `PodDisruptionBudget`. De même, si vous modifiez le module complémentaire pour utiliser une ancienne version du module complémentaire (rétrogradage) après la mise à niveau vers une version avec un `PodDisruptionBudget`, le `PodDisruptionBudget` n'est pas supprimé. Pour supprimer le `PodDisruptionBudget`, exécutez la commande suivante :

```
kubectl delete poddisruptionbudget coredns -n kube-system
```

- Dans les versions complémentaires d'EKS `v1.10.1-eksbuild.5` et versions ultérieures, modifiez la tolérance par défaut de `node-role.kubernetes.io/master:NoSchedule` à pour vous conformer `node-role.kubernetes.io/control-plane:NoSchedule` à KEP 2067. Pour plus d'informations sur KEP 2067, consultez KEP-2067: Reame the kubeadm "master" label and taint dans Kubernetes Enhancement Proposals (KEPs) sur GitHub.

  Dans les versions `v1.8.7-eksbuild.8` complémentaires d'EKS `v1.9.3-eksbuild.9` et les versions ultérieures, les deux tolérances sont définies pour être compatibles avec chaque Kubernetes version.

- Dans les versions complémentaires d'EKS `v1.9.3-eksbuild.11` `v1.10.1-eksbuild.7` et versions ultérieures, CoreDNS Deployment définit une valeur par défaut pour`topologySpreadConstraints`. La valeur par défaut garantit qu'CoreDNSPodsils sont répartis entre les zones de disponibilité s'il existe des nœuds disponibles dans plusieurs zones de disponibilité. Vous pouvez définir une valeur personnalisée qui sera utilisée à la place de la valeur par défaut. La valeur par défaut est la suivante :

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

CoreDNSconsidérations relatives **`1.11`** à la mise à niveau

- Dans les versions complémentaires d'EKS `v1.11.1-eksbuild.4` et versions ultérieures, l'image du conteneur est basée sur une image de base minimale gérée par Amazon EKS Distro, qui contient un minimum de packages et ne possède pas de shell. Pour plus d'informations, consultez Amazon EKS Distro. L'utilisation et le dépannage de l'CoreDNSimage restent les mêmes.

Création du module complémentaire Amazon EKS

Créez le type Amazon EKS du module complémentaire. Check

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.

1. Déterminez la version du module complémentaire actuellement installée sur votre cluster.

```
kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
d : -f 3
```

   L'exemple qui suit illustre un résultat.

```
v1.10.1-eksbuild.11
```

2. Déterminez le type de module complémentaire installé sur votre cluster. Selon l'outil avec lequel vous avez créé votre cluster, le type de module complémentaire Amazon EKS peut ne pas être actuellement installé sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

   Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster, et vous n'avez donc pas besoin de suivre les étapes restantes de cette procédure. Si une erreur est renvoyée, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Suivez les étapes restantes de cette procédure pour l'installer.

3. Enregistrez la configuration du module complémentaire actuellement installé.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

4. Créez le module complémentaire à l'aide de la AWS CLI. Si vous souhaitez utiliser le AWS Management Console ou `eksctl` pour créer le module complémentaire, consultez Création d'un module complémentaire et spécifiez `coredns` le nom du module complémentaire. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée.

   - Remplacez *my-cluster* par le nom de votre cluster.
   - Remplacez *v1.11.1-eksbuild.9* par la dernière version répertoriée dans le tableau des dernières versions pour la version de votre cluster.

```
aws eks create-addon --cluster-name my-cluster --addon-name coredns --addon-
version v1.11.1-eksbuild.9
```

Si vous avez appliqué à votre module complémentaire actuel des paramètres personnalisés qui entrent en conflit avec les paramètres par défaut du module complémentaire Amazon EKS, la création peut échouer. Si la création échoue, vous recevez un message d'erreur qui peut vous aider à résoudre le problème. Vous pouvez également ajouter **--resolve-conflicts OVERWRITE** à la commande précédente. Cela permet au module complémentaire de remplacer les paramètres personnalisés existants. Une fois que vous avez créé le module complémentaire, vous pouvez le mettre à jour avec vos paramètres personnalisés.

5. Assurez-vous que la dernière version du module complémentaire correspondant à la version de Kubernetes de votre cluster a été ajoutée à votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

La création du module complémentaire peut prendre plusieurs secondes.

L'exemple qui suit illustre un résultat.

```
v1.11.1-eksbuild.9
```

6. Si vous avez personnalisé les paramètres du module complémentaire, avant de créer le module complémentaire Amazon EKS, utilisez la configuration que vous avez enregistrée lors d'une étape précédente pour mettre à jour le module complémentaire Amazon EKS avec vos paramètres personnalisés.

Mise à jour du module complémentaire Amazon EKS

Mettez à jour le type de module complémentaire Amazon EKS. Si vous n'avez pas ajouté le type de module complémentaire Amazon EKS à votre cluster, ajoutez-le ou consultez la rubrique Mise à jour du module complémentaire autogéré au lieu de terminer cette procédure.

1. Déterminez la version du module complémentaire actuellement installée sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 "addon.addonVersion" --output text
```

L'exemple qui suit illustre un résultat.

```
v1.10.1-eksbuild.11
```

Si la version renvoyée est identique à la version de Kubernetes de votre cluster dans le tableau des dernières versions, cela signifie que la dernière version est déjà installée sur votre cluster, et vous n'avez donc pas besoin de terminer cette procédure. Si vous recevez une erreur au lieu d'un numéro de version dans votre sortie, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Vous devez créer le module complémentaire avant de pouvoir le mettre à jour à l'aide de cette procédure.

2. Enregistrez la configuration du module complémentaire actuellement installé.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

3. Mettez à jour votre module complémentaire à l'aide de la AWS CLI. Si vous souhaitez utiliser AWS Management Console ou mettre `eksctl` à jour le module complémentaire, consultezMise à jour d'un module complémentaire. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée.

   - Remplacez *my-cluster* par le nom de votre cluster.
   - Remplacez *v1.11.1-eksbuild.9* par la dernière version répertoriée dans le tableau des dernières versions pour la version de votre cluster.
   - L'option **--resolve-conflicts** *PRESERVE* conserve les valeurs de configuration existantes pour le module complémentaire. Si vous avez défini des valeurs personnalisées pour les paramètres des modules complémentaires et que vous n'utilisez pas cette option, Amazon EKS remplace vos valeurs par ses valeurs par défaut. Si vous utilisez cette option, nous vous recommandons de tester les modifications de champ et de valeur sur un cluster hors production avant de mettre à jour le module complémentaire sur votre cluster de production. Si vous remplacez cette valeur par `OVERWRITE`, tous les paramètres sont remplacés par les valeurs par défaut d'Amazon EKS. Si vous avez défini des valeurs personnalisées pour certains paramètres, il est possible qu'elles soient remplacées par les valeurs par défaut d'Amazon EKS. Si vous remplacez cette valeur par `none`, Amazon EKS ne modifie la valeur

d'aucun paramètre, mais la mise à jour risque d'échouer. Si la mise à jour échoue, vous recevez un message d'erreur pour vous aider à résoudre le conflit.

- Si vous ne mettez à jour aucun paramètre de configuration, supprimez **`--configuration-values '{`*`"replicaCount":3`*`}'`** de la commande. Si vous mettez à jour un paramètre de configuration, remplacez *`"replicaCount":3`* par le paramètre que vous voulez définir. Dans cet exemple, le nombre de réplicas de CoreDNS est défini sur 3. La valeur que vous spécifiez doit être valide pour le schéma de configuration. Si vous ne connaissez pas le schéma de configuration**`aws eks describe-addon-configuration --addon-name coredns --addon-version `***`v1.11.1-eksbuild.9`***, lancez-le en remplaçant *`v1.11.1-eksbuild.9`* par le numéro de version du module complémentaire dont vous souhaitez consulter la configuration. Le schéma est renvoyé dans la sortie. Si vous disposez déjà d'une configuration personnalisée et vous voulez la supprimer et rétablir les valeurs par défaut d'Amazon EKS pour tous les paramètres, supprimez *`"replicaCount":3`* de la commande, de sorte que le champ **`{}`** soit vide. Pour plus d'informations sur les paramètres CoreDNS, consultez la section Personnalisation du service DNS (français non garanti) dans la documentation Kubernetes.

```
aws eks update-addon --cluster-name my-cluster --addon-name coredns --addon-
version v1.11.1-eksbuild.9 \
    --resolve-conflicts PRESERVE --configuration-values '{"replicaCount":3}'
```

La mise à jour peut prendre plusieurs secondes.

4. Assurez-vous que la version du module complémentaire a été mise à jour. Remplacez *`my-cluster`* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns
```

La mise à jour peut prendre plusieurs secondes.

L'exemple qui suit illustre un résultat.

```
{
    "addon": {
        "addonName": "coredns",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.11.1-eksbuild.9",
        "health": {
```

```
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/coredns/
d2c34f06-1111-2222-1eb0-24f64ce37fa4",
        "createdAt": "2023-03-01T16:41:32.442000+00:00",
        "modifiedAt": "2023-03-01T18:16:54.332000+00:00",
        "tags": {},
        "configurationValues": "{\"replicaCount\":3}"
    }
}
```

Mise à jour du module complémentaire autogéré

> **Important**
>
> Nous recommandons d'ajouter le type Amazon EKS du module complémentaire à votre cluster au lieu d'utiliser le type autogéré du module complémentaire. Si la différence entre les types ne vous est pas familière, consultez the section called "Modules complémentaires Amazon EKS". Pour plus d'informations sur l'ajout d'un module complémentaire Amazon EKS à votre cluster, consultez the section called "Création d'un module complémentaire". Si vous ne parvenez pas à utiliser le module complémentaire Amazon EKS, nous vous encourageons à signaler les raisons pour lesquelles vous ne pouvez pas utiliser le module complémentaire Amazon EKS dans le GitHub référentiel de feuilles de route pour les conteneurs.

1. Vérifiez que le type autogéré du module complémentaire est installé sur votre cluster. Remplacez *my-cluster* par le nom de votre cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Si un message d'erreur est renvoyé, le type autogéré du module complémentaire est installé sur votre cluster. Suivez les étapes restantes de cette procédure. Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster. Pour mettre à jour le type Amazon EKS du module complémentaire, suivez la procédure décrite dans la rubrique Mise à jour du module complémentaire Amazon EKS plutôt que cette procédure. Si les différences entre les types de modules complémentaires ne vous sont pas familières, consultez Modules complémentaires Amazon EKS.

2. Découvrez quelle version de l'image de conteneur est actuellement installée sur votre cluster.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

L'exemple qui suit illustre un résultat.

```
v1.8.7-eksbuild.2
```

3. Si votre version actuelle de CoreDNS est `v1.5.0` ou ultérieure, mais qu'elle est antérieure à la version répertoriée dans le tableau Versions de CoreDNS, ignorez cette étape. Si votre version actuelle est antérieure à la version `1.5.0`, vous devez modifier la `ConfigMap` pour que CoreDNS utilise le module complémentaire, plutôt que le module complémentaire proxy.

   1. Ouvrez la carte de configuration avec la commande suivante.

```
kubectl edit configmap coredns -n kube-system
```

   2. Remplacez `proxy` dans la ligne suivante par `forward`. Enregistrez le fichier et quittez l'éditeur.

```
proxy . /etc/resolv.conf
```

4. Si vous avez initialement déployé votre cluster sur Kubernetes version `1.17` ou antérieure, vous devrez peut-être supprimer une ligne abandonnée de votre manifeste CoreDNS.

> **Important**
>
> Vous devez effectuer cette étape avant la mise à jour vers la version `1.7.0` de CoreDNS, mais il est recommandé d'effectuer cette étape même en cas de mise à jour vers une version antérieure.

   1. Vérifiez si votre manifeste CoreDNS possède la ligne.

```
kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
 grep upstream
```

Si aucune sortie n'est renvoyée, votre manifeste ne dispose pas de la ligne et vous pouvez passer à l'étape suivante pour mettre à jour CoreDNS. Si la sortie est renvoyée, vous devez supprimer la ligne.

2. Modifiez la `ConfigMap` en utilisant la commande suivante, en supprimant la ligne dans le fichier contenant le mot `upstream`. Ne modifiez rien d'autre dans le fichier. Une fois la ligne supprimée, enregistrez les modifications.

```
kubectl edit configmap coredns -n kube-system -o yaml
```

5. Récupérez la version de votre image CoreDNS actuelle :

```
kubectl describe deployment coredns -n kube-system | grep Image
```

L'exemple qui suit illustre un résultat.

```
602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.8.7-eksbuild.2
```

6. Si vous effectuez une mise à jour vers de CoreDNS vers la version `1.8.3` ou une version ultérieure, vous devez ajouter l'autorisation `endpointslices` au `system:coredns` Kubernetes `clusterrole`.

```
kubectl edit clusterrole system:coredns -n kube-system
```

Ajoutez les lignes suivantes sous les lignes d'autorisation existantes dans la section `rules` du fichier.

```
[...]
- apiGroups:
  - discovery.k8s.io
  resources:
  - endpointslices
  verbs:
  - list
  - watch
[...]
```

7. Mettez à jour le module complémentaire CoreDNS en remplaçant *602401143452* et *region-code* par les valeurs obtenues à l'étape précédente. Remplacez *v1.11.1-eksbuild.9* par la

version de CoreDNS répertoriée dans le tableau des dernières versions pour votre version de Kubernetes.

```
kubectl set image deployment.apps/coredns -n kube-system
 coredns=602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.11.1-
eksbuild.9
```

L'exemple qui suit illustre un résultat.

```
deployment.apps/coredns image updated
```

8. Vérifiez à nouveau la version d'image de conteneur pour confirmer qu'elle a été mise à jour vers la version que vous avez spécifiée à l'étape précédente.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

L'exemple qui suit illustre un résultat.

```
v1.11.1-eksbuild.9
```

Mise à l'échelle automatique CoreDNS

Lorsque vous lancez un cluster Amazon EKS avec au moins un nœud, deux répliques de l'CoreDNSimage sont déployées par défaut, quel que soit le nombre de nœuds déployés dans votre cluster. Deployment Les CoreDNS pods fournissent une résolution de nom pour tous les pods du cluster. Les applications utilisent la résolution de noms pour se connecter aux pods et aux services du cluster ainsi qu'aux services extérieurs au cluster. À mesure que le nombre de demandes de résolution de noms (requêtes) émanant des pods augmente, les CoreDNS pods peuvent être débordés et ralentir, et rejeter les demandes qu'ils ne peuvent pas traiter.

Pour gérer l'augmentation de la charge sur les CoreDNS pods, envisagez un système de mise à l'échelle automatique pourCoreDNS. Amazon EKS peut gérer le dimensionnement automatique du CoreDNS déploiement dans la version complémentaire EKS deCoreDNS. Cet CoreDNS autoscaler surveille en permanence l'état du cluster, notamment le nombre de nœuds et de cœurs de processeur. Sur la base de ces informations, le contrôleur adaptera dynamiquement le nombre de répliques du CoreDNS déploiement dans un cluster EKS. Cette fonctionnalité fonctionne pour toutes

les versions d'EKS CoreDNS `v1.9` `1.25` et les versions ultérieures. Pour plus d'informations sur les versions compatibles avec CoreDNS Autoscaling, consultez la section suivante.

Nous vous recommandons d'utiliser cette fonctionnalité conjointement avec d'autres bonnes pratiques d'EKS Cluster Autoscaling afin d'améliorer la disponibilité globale des applications et l'évolutivité du cluster.

Prérequis

Pour qu'Amazon EKS puisse étendre votre CoreDNS déploiement, trois conditions préalables sont requises :

- Vous devez utiliser la version complémentaire EKS deCoreDNS.
- Votre cluster doit exécuter au moins les versions de cluster et les versions de plate-forme minimales.
- Votre cluster doit exécuter au moins la version minimale du module complémentaire EKS deCoreDNS.

Version minimale du cluster

La mise à l'échelle automatique CoreDNS est effectuée par un nouveau composant dans le plan de contrôle du cluster, géré par Amazon EKS. Pour cette raison, vous devez mettre à niveau votre cluster vers une version d'EKS prenant en charge la version minimale de plate-forme contenant le nouveau composant.

Un nouveau cluster Amazon EKS. Pour en déployer un, consultez Démarrer avec Amazon EKS. Le cluster doit être d'une version Kubernetes `1.25` ou ultérieure. Le cluster doit exécuter l'une des Kubernetes versions et versions de plate-forme répertoriées dans le tableau suivant ou une version ultérieure. Toute version de Kubernetes et de plateforme ultérieure à celles répertoriées est également prise en charge. Vous pouvez vérifier votre version Kubernetes actuelle en remplaçant *`my-cluster`* dans la commande suivante par le nom de votre cluster et en exécutant la commande :

```
aws eks describe-cluster
              --name my-cluster --query cluster.version --output
              text
```

Version de Kubernetes	Version de plateforme
1.29.3	eks.7
1.28.8	eks.13
1.27.12	eks.17
1.26.15	eks.18
1.25.16	eks.19

> **Note**
>
> Toutes les versions de plate-forme des Kubernetes versions ultérieures sont également prises en charge, par exemple 1.30 eks.1 les Kubernetes versions ultérieures.

Version minimale du module complémentaire EKS

Version de Kubernetes	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**
	v1.11.1-eksbuild.9	v1.10.1-eksbuild.11	v1.10.1-eksbuild.11	v1.9.3-eksbuild.15	v1.9.3-eksbuild.15

Configuration de CoreDNS l'autoscaling dans AWS Management Console

1. Assurez-vous que votre cluster est égal ou supérieur à la version minimale du cluster.

   Amazon EKS met automatiquement à niveau les clusters entre les versions de plate-forme d'une même Kubernetes version, et vous ne pouvez pas démarrer ce processus vous-même. Au lieu de cela, vous pouvez mettre à niveau votre cluster vers la Kubernetes version suivante, et le cluster sera mis à niveau vers cette version K8s et la dernière version de la plateforme. Par exemple, si vous effectuez une mise à niveau depuis 1.25 vers1.26, le cluster sera mis à niveau vers1.26.15 eks.18.

Les nouvelles versions de Kubernetes introduisent parfois des modifications importantes. Par conséquent, nous vous recommandons de tester le comportement de vos applications en utilisant un cluster distinct de la nouvelle Kubernetes version avant de mettre à jour vos clusters de production.

Pour mettre à niveau un cluster vers une nouvelle Kubernetes version, suivez la procédure décrite dansMise à jour d'une version Kubernetes de cluster Amazon EKS.

2. Assurez-vous que vous disposez du module complémentaire EKS pour le déploiement autogéréCoreDNS, et non pour le CoreDNS déploiement autogéré.

   Selon l'outil avec lequel vous avez créé votre cluster, le type de module complémentaire Amazon EKS peut ne pas être actuellement installé sur votre cluster. Pour savoir quel type de module complémentaire est installé sur votre cluster, vous pouvez exécuter la commande suivante. Remplacez `my-cluster` par le nom de votre cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Si un numéro de version est renvoyé, le type Amazon EKS du module complémentaire est installé sur votre cluster et vous pouvez passer à l'étape suivante. Si une erreur est renvoyée, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Effectuez les étapes restantes de la procédure Création du module complémentaire Amazon EKS pour remplacer la version autogérée par le module complémentaire Amazon EKS.

3. Assurez-vous que la version de votre module complémentaire EKS pour CoreDNS est identique ou supérieure à la version minimale du module complémentaire EKS.

   Déterminez la version du module complémentaire actuellement installée sur votre cluster. Vous pouvez enregistrer AWS Management Console ou exécuter la commande suivante :

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   L'exemple qui suit illustre un résultat.

   ```
   v1.10.1-eksbuild.11
   ```

Comparez cette version avec la version minimale du module complémentaire EKS dans la section précédente. Si nécessaire, mettez à niveau le module complémentaire EKS vers une version supérieure en suivant la procédureMise à jour du module complémentaire Amazon EKS.

4. Ajoutez la configuration de mise à l'échelle automatique aux paramètres de configuration facultatifs du module complémentaire EKS.

   a. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

   b. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire.

   c. Choisissez l'onglet Modules complémentaires.

   d. Cochez la case en haut à droite de la zone du CoreDNS module complémentaire, puis choisissez Modifier.

   e. Sur la CoreDNS page de configuration :

      i. Sélectionnez la version que vous souhaitez utiliser. Nous vous recommandons de conserver la même version que celle de l'étape précédente et de mettre à jour la version et la configuration dans le cadre d'actions distinctes.

      ii. Sélectionnez Paramètres de configuration facultatifs.

      iii. Entrez la clé JSON `"autoscaling":` et la valeur d'un objet JSON imbriqué avec une clé `"enabled":` et une valeur `true` dans Valeurs de configuration. Le texte obtenu doit être un objet JSON valide. Si cette clé et cette valeur sont les seules données de la zone de texte, entourez-les d'accolades `{}`. L'exemple suivant montre que la mise à l'échelle automatique est activée :

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

      iv. (Facultatif) Vous pouvez fournir des valeurs minimales et maximales auxquelles la mise à l'échelle automatique peut redimensionner le nombre de CoreDNS pods.

      L'exemple suivant montre que la mise à l'échelle automatique est activée et que toutes les clés facultatives ont des valeurs. Nous recommandons que le nombre minimum de

CoreDNS pods soit toujours supérieur à 2 pour garantir la résilience du service DNS du cluster.

```
{
  "autoScaling": {
    "enabled": true,
    "minReplicas": 2,
    "maxReplicas": 10
  }
}
```

f. Pour appliquer la nouvelle configuration en remplaçant les CoreDNS modules, choisissez Enregistrer les modifications.

Amazon EKS applique les modifications aux modules complémentaires EKS en Kubernetesdéployant le déploiement pour CoreDNS. Vous pouvez suivre l'état du déploiement dans l'historique des mises à jour du module complémentaire dans AWS Management Console et avec`kubectl rollout status deployment/coredns --namespace kube-system`.

`kubectl rollout`possède les commandes suivantes :

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Si le déploiement prend trop de temps, Amazon EKS annulera le déploiement et un message indiquant le type de mise à jour de l'extension et le statut Echec sera ajouté à l'historique des mises à jour de l'extension. Pour étudier les problèmes éventuels, commencez par l'historique du déploiement et exécutez `kubectl logs` sur un CoreDNS module pour consulter les journaux deCoreDNS.

5. Si le statut de la nouvelle entrée dans l'historique des mises à jour est Réussi, le déploiement est terminé et le module complémentaire utilise la nouvelle configuration dans tous les CoreDNS

modules. Lorsque vous modifiez le nombre de nœuds et de cœurs de processeur des nœuds du cluster, Amazon EKS adapte le nombre de répliques du CoreDNS déploiement.

Configuration de CoreDNS l'autoscaling dans AWS Command Line Interface

1. Assurez-vous que votre cluster est égal ou supérieur à la version minimale du cluster.

   Amazon EKS met automatiquement à niveau les clusters entre les versions de plate-forme d'une même Kubernetes version, et vous ne pouvez pas démarrer ce processus vous-même. Au lieu de cela, vous pouvez mettre à niveau votre cluster vers la Kubernetes version suivante, et le cluster sera mis à niveau vers cette version K8s et la dernière version de la plateforme. Par exemple, si vous effectuez une mise à niveau depuis `1.25` vers`1.26`, le cluster sera mis à niveau vers`1.26.15 eks.18`.

   Les nouvelles versions de Kubernetes introduisent parfois des modifications importantes. Par conséquent, nous vous recommandons de tester le comportement de vos applications en utilisant un cluster distinct de la nouvelle Kubernetes version avant de mettre à jour vos clusters de production.

   Pour mettre à niveau un cluster vers une nouvelle Kubernetes version, suivez la procédure décrite dansMise à jour d'une version Kubernetes de cluster Amazon EKS.
2. Assurez-vous que vous disposez du module complémentaire EKS pour le déploiement autogéréCoreDNS, et non pour le CoreDNS déploiement autogéré.

   Selon l'outil avec lequel vous avez créé votre cluster, le type de module complémentaire Amazon EKS peut ne pas être actuellement installé sur votre cluster. Pour savoir quel type de module complémentaire est installé sur votre cluster, vous pouvez exécuter la commande suivante. Remplacez `my-cluster` par le nom de votre cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster. Si une erreur est renvoyée, cela signifie que le type de module complémentaire Amazon EKS n'est pas installé sur votre cluster. Effectuez les étapes restantes de la procédure Création du module complémentaire Amazon EKS pour remplacer la version autogérée par le module complémentaire Amazon EKS.

3. Assurez-vous que la version de votre module complémentaire EKS pour CoreDNS est identique ou supérieure à la version minimale du module complémentaire EKS.

   Déterminez la version du module complémentaire actuellement installée sur votre cluster. Vous pouvez enregistrer AWS Management Console ou exécuter la commande suivante :

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   L'exemple qui suit illustre un résultat.

   ```
   v1.10.1-eksbuild.11
   ```

   Comparez cette version avec la version minimale du module complémentaire EKS dans la section précédente. Si nécessaire, mettez à niveau le module complémentaire EKS vers une version supérieure en suivant la procédureMise à jour du module complémentaire Amazon EKS.

4. Ajoutez la configuration de mise à l'échelle automatique aux paramètres de configuration facultatifs du module complémentaire EKS.

   Exécutez la AWS CLI commande suivante. Remplacez `my-cluster` par le nom de votre cluster et l'ARN du rôle IAM par le rôle que vous utilisez.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true}}'
   ```

   Amazon EKS applique les modifications aux modules complémentaires EKS en Kubernetesdéployant le déploiement pour CoreDNS. Vous pouvez suivre l'état du déploiement dans l'historique des mises à jour du module complémentaire dans AWS Management Console et avec`kubectl rollout status deployment/coredns --namespace kube-system`.

   `kubectl rollout`possède les commandes suivantes :

   ```
   kubectl rollout

   history  -- View rollout history
   pause    -- Mark the provided resource as paused
   ```

```
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Si le déploiement prend trop de temps, Amazon EKS annulera le déploiement et un message indiquant le type de mise à jour de l'extension et le statut Echec sera ajouté à l'historique des mises à jour de l'extension. Pour étudier les problèmes éventuels, commencez par l'historique du déploiement et exécutez `kubectl logs` sur un CoreDNS module pour consulter les journaux deCoreDNS.

5. (Facultatif) Vous pouvez fournir des valeurs minimales et maximales auxquelles la mise à l'échelle automatique peut redimensionner le nombre de CoreDNS pods.

   L'exemple suivant montre que la mise à l'échelle automatique est activée et que toutes les clés facultatives ont des valeurs. Nous recommandons que le nombre minimum de CoreDNS pods soit toujours supérieur à 2 pour garantir la résilience du service DNS du cluster.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true}, "minReplicas": 2, "maxReplicas": 10}'
   ```

6. Vérifiez l'état de la mise à jour du module complémentaire en exécutant la commande suivante :

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns \
   ```

   Si vous voyez cette ligne :`"status": "ACTIVE"`, le déploiement est terminé et le module complémentaire utilise la nouvelle configuration dans tous les CoreDNS modules. Lorsque vous modifiez le nombre de nœuds et de cœurs de processeur des nœuds du cluster, Amazon EKS adapte le nombre de répliques du CoreDNS déploiement.

Métriques CoreDNS

CoreDNS en tant que module complémentaire EKS expose les métriques de CoreDNS sur le port 9153 au format Prometheus dans le service `kube-dns`. Vous pouvez utiliser Prometheus, l'agent Amazon CloudWatch ou tout autre système compatible pour récupérer (collecter) ces métriques.

Pour un exemple de configuration de récupération compatible à la fois avec Prometheus et l'agent CloudWatch, consultez Configuration de l'agent CloudWatch pour Prometheus dans le Guide de l'utilisateur Amazon CloudWatch.

Utilisation du module complémentaire Kubernetes **kube-proxy**

> **Important**
>
> Nous recommandons d'ajouter le type Amazon EKS du module complémentaire à votre cluster au lieu d'utiliser le type autogéré du module complémentaire. Si la différence entre les types ne vous est pas familière, consultez the section called "Modules complémentaires Amazon EKS". Pour plus d'informations sur l'ajout d'un module complémentaire Amazon EKS à votre cluster, consultez the section called "Création d'un module complémentaire". Si vous ne parvenez pas à utiliser le module complémentaire Amazon EKS, nous vous encourageons à signaler les raisons pour lesquelles vous ne pouvez pas utiliser le module complémentaire Amazon EKS dans le GitHub référentiel de feuilles de route pour les conteneurs.

Le module complémentaire `kube-proxy` est déployé sur chaque nœud Amazon EC2 de votre cluster Amazon EKS. Il maintient les règles du réseau sur vos nœuds et permet la communication du réseau à vos Pods. Le module complémentaire n'est pas déployé sur les nœuds Fargate de votre cluster. Pour plus d'informations, consultez la section `kube-proxy`dans la documentation Kubernetes.

Le tableau suivant répertorie la dernière version du module complémentaire Amazon EKS pour chaque version de Kubernetes.

Version de Kubernetes	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
	v1.30.0-eksbuild	v1.29.3-eksbuild	v1.28.8-eksbuild	v1.27.16-eksbuild.5	v1.26.15-eksbuild.5	v1.25.16-eksbuild.8	v1.24.17-eksbuild.8	v1.23.17-eksbuild.9

> **Important**
>
> Une version antérieure de la documentation était incorrecte. `kube-proxy`versions `v1.28.5``v1.27.9`, et `v1.26.12` ne sont pas disponibles.
> Si vous gérez vous-même ce module complémentaire, les versions répertoriées dans le tableau peuvent ne pas être les mêmes que les versions autogérées disponibles.

Il existe deux types d'images de conteneur `kube-proxy` disponibles pour chaque version de cluster Amazon EKS :

- Default (Par défaut) : cette image est basée sur une image Docker basée sur Debian qui est gérée par la communauté Kubernetes en amont.
- Minimal (Minimale) : cette image est basée sur une image de base minimale gérée par Amazon EKS Distro, qui contient un package minimum sans aucun Shell. Pour plus d'informations, consultez Amazon EKS Distro.

Dernière version disponible d'image de conteneur **`kube-proxy`** autogéré pour chaque version de cluster Amazon EKS

Type d'image	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
`kube-proxy` (type par défaut)	Seul le type minimal est disponible	Seul le type minimal est disponible	Seul le type minimal est disponible	Seul le type minimal est disponible	Seul le type minimal est disponible	Seul le type minimal est disponible	`v1.24.1-eksbuild.2`	`v1.23.16-eksbuild.2`
`kube-proxy` (type minimal)	`v1.30.0-minimal-eksbuild.`	`v1.29.3-minimal-eksbuild.`	`v1.28.8-minimal-eksbuild.`	`v1.27.1-minimal-eksbuild`	`v1.26.1-minimal-eksbuild`	`v1.25.1-minimal-eksbuild`	`v1.24.1-minimal-eksbuild`	`v1.23.17-minimal-eksbuild.5`

> **Important**
>
> - Le type d'image par défaut n'est pas disponible pour Kubernetes version `1.25` et versions ultérieures. Vous devez utiliser le type d'image minimal.
> - Lorsque vous mettez à jour un type de module complémentaire Amazon EKS, vous spécifiez une version valide du module complémentaire Amazon EKS, qui peut ne pas être répertoriée dans ce tableau. Cela est dû au fait que les versions du module complémentaire Amazon EKS ne correspondent pas toujours aux versions d'image de conteneur spécifiées lors de la mise à jour du type autogéré de ce module complémentaire. Lorsque vous mettez à jour le type autogéré de ce module complémentaire, vous spécifiez une version d'image de conteneur valide répertoriée dans ce tableau.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.

Considérations

- `Kube-proxy` sur un cluster Amazon EKS possède la même politique de compatibilité et d'inclinaison que Kubernetes. Découvrez comment Récupérez la compatibilité des versions de l'addon.
- `Kube-proxy` doit correspondre à la même version mineure que `kubelet` sur vos nœuds Amazon EC2.
- `Kube-proxy` ne peut pas être postérieur à la version mineure du plan de contrôle de votre cluster.
- Si vous avez récemment mis à jour votre cluster vers une nouvelle version mineure de Kubernetes, mettez alors à jour vos nœuds Amazon EC2 vers la même version mineure avant de mettre à jour `kube-proxy` vers la même version mineure que vos nœuds.

Mise à jour du module complémentaire autogéré **`kube-proxy`**

1. Vérifiez que le type autogéré du module complémentaire est installé sur votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name kube-proxy --query
 addon.addonVersion --output text
```

Si un message d'erreur est renvoyé, le type autogéré du module complémentaire est installé sur votre cluster. Les étapes restantes de cette rubrique concernent la mise à jour du type autogéré du module complémentaire. Si un numéro de version est renvoyé, le type de module complémentaire Amazon EKS est installé sur votre cluster. Pour le mettre à jour, utilisez la procédure décrite dans Mise à jour d'un module complémentaire, plutôt que celle décrite dans cette rubrique. Si les différences entre les types de modules complémentaires ne vous sont pas familières, consultez Modules complémentaires Amazon EKS.

2. Découvrez quelle version de l'image de conteneur est actuellement installée sur votre cluster.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image
```

L'exemple qui suit illustre un résultat.

```
Image:    602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.29.1-
eksbuild.2
```

Dans l'exemple de sortie, *v1.29.1-eksbuild.2* est la version installée sur le cluster.

3. Mettez à jour le module complémentaire kube-proxy en remplaçant *602401143452* et *region-code* par les valeurs de votre sortie à l'étape précédente. Remplacez *v1.30.0-eksbuild.3* par la version de kube-proxy répertoriée dans le tableau Dernière version d'image de conteneur kube-proxy disponible pour chaque version de cluster Amazon EKS. Vous pouvez spécifier un numéro de version pour la version par défautou le type minimal d'image.

```
kubectl set image daemonset.apps/kube-proxy -n kube-system kube-
proxy=602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.30.0-
eksbuild.3
```

L'exemple qui suit illustre un résultat.

```
daemonset.apps/kube-proxy image updated
```

4. Vérifiez que la nouvelle version est maintenant installée sur votre cluster.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image | cut -d ":" -f 3
```

L'exemple qui suit illustre un résultat.

```
v1.30.0-eksbuild.3
```

5. Si vous utilisez des nœuds x86 et Arm dans le même cluster et que votre cluster a été déployé avant le 17 août 2020. Ensuite, modifiez votre manifeste kube-proxy pour inclure un sélecteur de noeud pour plusieurs architectures matérielles avec la commande suivante. Il s'agit d'une opération ponctuelle. Une fois que vous avez ajouté le sélecteur à votre manifeste, vous n'avez pas besoin de l'ajouter chaque fois que vous mettez à jour le module complémentaire. Si votre cluster a été déployé à partir du 17 août 2020, kube-proxy est déjà compatible avec les architectures multiples.

   ```
   kubectl edit -n kube-system daemonset/kube-proxy
   ```

   Ajoutez le sélecteur de nœuds suivant au fichier dans l'éditeur, puis enregistrez le fichier. Pour obtenir un exemple d'inclusion de ce texte dans l'éditeur, consultez le fichier manifeste CNI sur GitHub. Cela permet à Kubernetes d'extraire l'image matérielle correcte en fonction de l'architecture matérielle du nœud.

   ```
   - key: "kubernetes.io/arch"
     operator: In
     values:
     - amd64
     - arm64
   ```

6. Si votre cluster a été créé à l'origine avec Kubernetes version 1.14 ou ultérieure, vous pouvez ignorer cette étape, car kube-proxy inclut déjà cette Affinity Rule. Si vous avez créé à l'origine un cluster Amazon EKS avec Kubernetes version 1.13 ou antérieure et que vous avez l'intention d'utiliser des nœuds Fargate dans votre cluster, modifiez votre manifeste kube-proxy pour y inclure une règle NodeAffinity empêchant les Pods kube-proxy de se programmer sur les nœuds Fargate. Il s'agit d'une modification unique. Une fois que vous avez ajouté Affinity Rule à votre manifeste, vous n'avez pas besoin de l'ajouter chaque fois que vous mettez à niveau votre cluster. Modifiez votre DaemonSet kube-proxy.

   ```
   kubectl edit -n kube-system daemonset/kube-proxy
   ```

Ajoutez la commande `Affinity Rule` ci-après à la section `DaemonSetspec` du fichier dans l'éditeur, puis enregistrez le fichier. Pour obtenir un exemple d'inclusion de ce texte dans l'éditeur, consultez le fichier manifeste CNI sur GitHub.

```
- key: eks.amazonaws.com/compute-type
  operator: NotIn
  values:
  - fargate
```

Accédez à Amazon Elastic Kubernetes Service à l'aide d'un point de terminaison d'interface (AWS PrivateLink)

Vous pouvez l'utiliser AWS PrivateLink pour créer une connexion privée entre votre VPC et Amazon Elastic Kubernetes Service. Vous pouvez accéder à Amazon EKS comme s'il se trouvait dans votre VPC, sans passer par une passerelle Internet, un appareil NAT, une connexion VPN ou AWS Direct Connect une connexion. Les instances de votre VPC ne nécessitent pas d'adresses IP publiques pour accéder à Amazon EKS.

Vous établissez cette connexion privée en créant un point de terminaison d'interface optimisé par AWS PrivateLink. Nous créons une interface réseau de point de terminaison dans chaque sous-réseau que vous activez pour le point de terminaison d'interface. Il s'agit d'interfaces réseau gérées par le demandeur qui servent de point d'entrée pour le trafic destiné à Amazon EKS.

Pour plus d'informations, consultez Accès aux Services AWS via AWS PrivateLink dans le Guide AWS PrivateLink .

Considérations relatives à Amazon EKS

- Avant de configurer un point de terminaison d'interface pour Amazon EKS, consultez Considérations dans le Guide AWS PrivateLink .
- Amazon EKS prend en charge les appels vers toutes ses actions d'API via le point de terminaison d'interface, mais pas vers les API Kubernetes. Le serveur d'API Kubernetes prend déjà en charge un point de terminaison privé. Le point de terminaison privé du serveur d'API Kubernetes crée un point de terminaison privé pour le serveur d'API Kubernetes que vous utilisez afin de communiquer avec votre cluster (à l'aide d'outils de gestion Kubernetes comme `kubectl`). Vous pouvez activer l'accès privé au serveur d'KubernetesAPI afin que toutes les communications entre vos nœuds

et le serveur d'API restent au sein de votre VPC. AWS PrivateLink car l'API Amazon EKS vous permet d'appeler les API Amazon EKS depuis votre VPC sans exposer le trafic à l'Internet public.

- Vous ne pouvez pas configurer Amazon EKS pour qu'il soit uniquement accessible via un point de terminaison d'interface.
- La tarification standard AWS PrivateLink s'applique aux points de terminaison d'interface pour Amazon EKS. Vous êtes facturé pour chaque heure d'approvisionnement de point de terminaison d'interface dans chaque zone de disponibilité, et pour les données traitées via le point de terminaison d'interface. Pour en savoir plus, consultez AWS PrivateLink Tarification.
- Les stratégies de point de terminaison de VPC sont prises en charge pour Amazon EKS. Par défaut, l'accès complet à Amazon EKS est autorisé via le point de terminaison d'interface. Vous pouvez également associer un groupe de sécurité aux interfaces réseau du point de terminaison afin de contrôler le trafic vers Amazon EKS via le point de terminaison d'interface.
- Vous pouvez utiliser les journaux de flux VPC pour capturer des informations sur le trafic IP entrant et sortant des interfaces réseau, y compris des points de terminaison d'interface. Vous pouvez publier les données du journal de flux sur Amazon CloudWatch ou Amazon S3. Pour de plus amples informations, consultez Journalisation du trafic IP à l'aide des journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.
- Vous pouvez accéder aux API Amazon EKS depuis un centre de données sur site en le connectant à un VPC doté d'un point de terminaison d'interface. Vous pouvez utiliser AWS Direct Connect ou AWS Site-to-Site VPN connecter vos sites locaux à un VPC.
- Vous pouvez connecter d'autres VPC au VPC avec un point de terminaison d'interface à l'aide d'un appairage AWS Transit Gateway ou de VPC. L'appairage de VPC est une connexion réseau entre deux VPC. Vous pouvez aussi établir une connexion d'appairage de VPC entre vos VPC, ou avec un VPC situé dans un autre compte. Les VPC peuvent être différents. Régions AWS Le trafic entre les VPC homologues reste sur le AWS réseau. Le trafic ne transite pas par l'Internet public. Une passerelle de transit est un hub de transit de réseau que vous pouvez utiliser pour relier vos VPC. Le trafic entre un VPC et un passerelle de transit reste sur le réseau mondial privé AWS . Le trafic n'est pas exposé à l'Internet public.
- Les points de terminaison d'interface d'un VPC pour Amazon EKS sont uniquement accessibles via`IPv4`. `IPv6` n'est pas pris en charge.
- AWS PrivateLink le support n'est pas disponible en Asie-Pacifique (Hyderabad), en Asie-Pacifique (Melbourne), en Asie-Pacifique (Osaka), au Canada Ouest (Calgary), en Europe (Espagne), en Europe (Zurich) ou au Moyen-Orient (Émirats arabes unis). Régions AWS

Créer un point de terminaison d'interface pour Amazon EKS

Vous pouvez créer un point de terminaison d'interface pour Amazon EKS à l'aide de la console Amazon VPC ou du AWS Command Line Interface ()AWS CLI. Pour plus d'informations, consultez Créer un point de terminaison d'un VPC dans le Guide AWS PrivateLink .

Création d'un point de terminaison d'interface pour Amazon EKS à l'aide du nom de service suivant :

```
com.amazonaws.region-code.eks
```

La fonctionnalité de DNS privé est activée par défaut lors de la création d'un point de terminaison d'interface pour Amazon EKS et d'autres Services AWS. Toutefois, vous devez vous assurer que les attributs de VPC suivants sont définis sur `true` : `enableDnsHostnames` et `enableDnsSupport`. Pour plus d'informations, consultez Affichage et mise à jour des attributs DNS pour votre VPC dans le Guide de l'utilisateur Amazon VPC. Lorsque la fonctionnalité de DNS privé est activée pour le point de terminaison d'interface :

- Vous pouvez envoyer n'importe quelle demande d'API à Amazon EKS en utilisant son nom DNS régional par défaut. Par exemple, `eks.region.amazonaws.com`. Pour obtenir une liste des API, consultez la Actions dans la Référence d'API Amazon EKS.
- Vous n'avez pas besoin d'apporter de modifications aux applications qui appellent les API EKS.
- Tout appel effectué vers le point de terminaison du service par défaut Amazon EKS est automatiquement acheminé via le point de terminaison de l'interface via le AWS réseau privé.

Charges de travail

Vos charges de travail sont déployées dans des conteneurs, qui sont déployés dans des Pods dans Kubernetes. Un Pod comprend un ou plusieurs conteneurs. En règle générale, un ou plusieurs Pods qui fournissent le même service sont déployés dans un service Kubernetes. Une fois que vous avez déployé plusieurs Pods qui fournissent le même service, vous pouvez :

- Afficher des informations sur les applications s'exécutant sur chacun de vos clusters à l'aide de AWS Management Console.
- Augmenter ou diminuer verticalement les Pods avec le Vertical Pod Autoscaler (VPA) de Kubernetes.
- Augmenter ou diminuer horizontalement le nombre de Pods nécessaires pour répondre à la demande avec le Horizontal Pod Autoscaler de Kubernetes.
- Créer un Network Load Balancer externe (pour les Pods accessibles depuis Internet) ou interne (pour les Pods privés) afin d'équilibrer le trafic réseau entre les Pods. L'équilibreur de charge achemine le trafic à la couche 4 du modèle OSI.
- Création d'un Répartition de la charge des applications sur Amazon EKS pour équilibrer le trafic des applications entre les Pods. L'Application Load Balancer achemine le trafic à la couche 7 du modèle OSI.
- Si vous débutez avec Kubernetes, cette rubrique vous aide avec le Déployer un exemple d'application .
- Vous pouvez restreindre les adresses IP qui peuvent être affectées à un service avec `externalIPs`.

Déployer un exemple d'application

Dans cette rubrique, vous déployez un exemple d'application sur votre cluster.

Prérequis

- Un cluster Kubernetes existant avec au moins un nœud. Si vous n'avez pas de cluster Amazon EKS existant, vous pouvez en déployer un en utilisant l'un des guides Démarrer avec Amazon EKS. Si vous déployez une application Windows, vous devez avoir la prise en charge de Windows activée pour votre cluster et au moins un nœud Amazon EC2 Windows.

- `Kubectl` installé sur votre ordinateur. Pour plus d'informations, consultez Installation ou mise à jour de `kubectl`.
- `Kubectl` configuré pour communiquer avec votre cluster. Pour plus d'informations, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.
- Si vous prévoyez de déployer votre charge de travail d'exemple sur Fargate, vous devez disposer d'un profil Fargate existant qui inclut le même espace de noms créé dans ce didacticiel, à savoir `eks-sample-app`, à moins que vous ne changiez le nom. Si vous avez utilisé l'un des guides de démarrage pour créer votre cluster, vous devrez créer un nouveau profil ou ajouter l'espace de noms à votre profil existant, car le profil créé dans les guides de démarrage ne spécifie pas l'espace de noms utilisé dans ce didacticiel. Votre VPC doit également disposer d'au moins un sous-réseau privé.

Pour déployer un exemple d'application

Bien que de nombreuses variables soient modifiables dans les étapes suivantes, nous vous recommandons de ne modifier les valeurs des variables qu'aux endroits spécifiés. Une fois que vous aurez une meilleure compréhension des Pods, des déploiements et des services Kubernetes, vous pourrez expérimenter la modification d'autres valeurs.

1. Créez un espace de noms . Un espace de noms vous permet de regrouper des ressources dans Kubernetes. Pour plus d'informations, consultez Espaces de noms dans la documentation Kubernetes. Si vous prévoyez de déployer votre application d'exemple vers AWS Fargate, assurez-vous que la valeur de `namespace` dans votre AWS Fargate profil est `eks-sample-app`.

   ```
   kubectl create namespace eks-sample-app
   ```

2. Créez un déploiement Kubernetes. Cet exemple de déploiement extrait une image de conteneur d'un référentiel public et en déploie trois répliques (Pods individuels) sur votre cluster. Pour en savoir plus, consultez Déploiements dans la documentation Kubernetes. Vous pouvez déployer l'application sur des nœuds Linux ou Windows. Si vous déployez sur Fargate, vous ne pouvez déployer qu'une application Linux.

   a. Enregistrez le contenu suivant dans un fichier nommé `eks-sample-deployment.yaml`. Les conteneurs de l'application d'exemple n'utilisent pas de stockage réseau, mais vous pouvez avoir des applications qui en ont besoin. Pour plus d'informations, consultez Stockage.

Linux

Le `amd64` ou `arm64` `values` sous la clé `kubernetes.io/arch` signifient que l'application peut être déployée sur l'une ou l'autre architecture matérielle (si vous avez les deux dans votre cluster). Cela est possible car cette image est une image multi-architecture, mais toutes ne le sont pas. Vous pouvez déterminer l'architecture matérielle prise en charge par l'image en consultant les détails de l'image dans le référentiel d'où vous la retirez. Lorsque vous déployez des images qui ne prennent pas en charge un type d'architecture matérielle, ou pour lesquelles vous ne souhaitez pas que l'image soit déployée, supprimez ce type du manifeste. Pour plus d'informations, consultez Étiquettes, annotations et rejets connus dans la documentation Kubernetes.

Le `nodeSelector` de `kubernetes.io/os: linux` signifie que si vous possédez des nœuds Linux et Windows (par exemple) dans votre cluster, l'image ne sera déployée que sur des nœuds Linux. Pour plus d'informations, consultez Étiquettes, annotations et rejets connus dans la documentation Kubernetes.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-linux-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-linux-app
  template:
    metadata:
      labels:
        app: eks-sample-linux-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: kubernetes.io/arch
```

```
                operator: In
                values:
                - amd64
                - arm64
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
      nodeSelector:
        kubernetes.io/os: linux
```

Windows

Le `nodeSelector` de `kubernetes.io/os: windows` signifie que si vous possédez des nœuds Windows et Linux (par exemple) dans votre cluster, l'image ne sera déployée que sur des nœuds Windows. Pour plus d'informations, consultez Étiquettes, annotations et rejets connus dans la documentation Kubernetes.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-windows-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-windows-app
  template:
    metadata:
      labels:
        app: eks-sample-windows-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
```

```
                - key: beta.kubernetes.io/arch
                  operator: In
                  values:
                  - amd64
      containers:
      - name: windows-server-iis
        image: mcr.microsoft.com/windows/servercore:ltsc2019
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
        command:
        - powershell.exe
        - -command
        - "Add-WindowsFeature Web-Server; Invoke-WebRequest -UseBasicParsing
 -Uri 'https://dotnetbinaries.blob.core.windows.net/servicemonitor/2.0.1.6/
ServiceMonitor.exe' -OutFile 'C:\\ServiceMonitor.exe'; echo
 '<html><body><br/><br/><marquee><H1>Hello EKS!!!<H1><marquee></body><html>'
 > C:\\inetpub\\wwwroot\\default.html; C:\\ServiceMonitor.exe 'w3svc'; "
      nodeSelector:
        kubernetes.io/os: windows
```

b. Appliquer le manifeste de déploiement à votre cluster.

```
kubectl apply -f eks-sample-deployment.yaml
```

3. Créer un service. Un service vous permet d'accéder à toutes les réplicas via une seule adresse IP ou un seul nom. Pour plus d'informations, consultez Service dans la documentation Kubernetes. Bien que cela ne soit pas implémenté dans l'exemple d'application, si certaines applications doivent interagir avec d'autres AWS services, nous vous recommandons de créer des comptes de Kubernetes service pour votre Pods compte et de les associer à des comptes AWS IAM. En spécifiant des comptes de service, vos Pods ne disposent que des autorisations minimales que vous spécifiez pour qu'ils puissent interagir avec d'autres services. Pour plus d'informations, consultez Rôles IAM pour les comptes de service.

   a. Enregistrez le contenu suivant dans un fichier nommé `eks-sample-service.yaml`. Kubernetes attribue au service sa propre adresse IP qui n'est accessible que depuis l'intérieur du cluster. Pour accéder au service depuis l'extérieur de votre cluster, déployez l'AWS Load Balancer Controller pour équilibrer la charge du trafic d'application ou de réseau vers le service.

Linux

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-linux-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  selector:
    app: eks-sample-linux-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Windows

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-windows-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  selector:
    app: eks-sample-windows-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

b. Appliquer le manifeste du service à votre cluster.

```
kubectl apply -f eks-sample-service.yaml
```

4. Affichez toutes les ressources présentes dans l'espace de noms `eks-sample-app`.

```
kubectl get all -n eks-sample-app
```

L'exemple qui suit illustre un résultat.

Si vous avez déployé des ressources Windows, toutes les instances de *linux* dans la sortie suivante sont windows. Les autres *exemples de valeurs* peuvent être différents de votre sortie.

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/eks-sample-linux-deployment-65b7669776-m6qxz     1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-mmxvd     1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-qzn22     1/1     Running   0          27m

NAME                                 TYPE         CLUSTER-IP      EXTERNAL-IP
 PORT(S)   AGE
service/eks-sample-linux-service     ClusterIP    10.100.74.8     <none>        80/
TCP    32m

NAME                                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/eks-sample-linux-deployment    3/3     3            3           27m

NAME                                                      DESIRED   CURRENT   READY
   AGE
replicaset.apps/eks-sample-linux-deployment-776d8f8fd8    3         3         3
   27m
```

Dans la sortie, vous voyez le service et le déploiement spécifiés dans l'exemple de manifestes déployés dans les étapes précédentes. Vous voyez aussi trois Pods. Ceci est dû au fait que 3 replicas ont été spécifiés dans l'exemple de manifeste. Pour plus d'informations sur les Pods, consultez la rubrique Pods dans la documentation Kubernetes. Kubernetes crée automatiquement la ressource replicaset, même si cela n'est pas spécifié dans l'exemple de manifestes. Pour plus d'informationsReplicaSets, consultez ReplicaSetla Kubernetes documentation.

> **Note**
>
> Kubernetes conserve le nombre de réplicas spécifié dans le manifeste. S'il s'agissait d'un déploiement de production et que vous souhaitiez que Kubernetes mette à l'échelle horizontalement le nombre de réplicas ou qu'il mette à l'échelle verticalement les

ressources de calcul pour les Pods, pour ce faire, utilisez le Horizontal Pod Autoscaler et le Vertical Pod Autoscaler (VPA).

5. Affichez les détails du service déployé. Si vous avez déployé un service Windows, remplacez *linux* par **windows**.

```
kubectl -n eks-sample-app describe service eks-sample-linux-service
```

L'exemple qui suit illustre un résultat.

Si vous avez déployé des ressources Windows, toutes les instances de *linux* dans la sortie suivante sont `windows`. Les autres *exemples de valeurs* peuvent être différents de votre sortie.

```
Name:              eks-sample-linux-service
Namespace:         eks-sample-app
Labels:            app=eks-sample-linux-app
Annotations:       <none>
Selector:          app=eks-sample-linux-app
Type:              ClusterIP
IP Families:       <none>
IP:                10.100.74.8
IPs:               10.100.74.8
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         192.168.24.212:80,192.168.50.185:80,192.168.63.93:80
Session Affinity:  None
Events:            <none>
```

Dans la sortie précédente, la valeur pour `IP:` est une adresse IP unique qui peut être atteinte à partir de n'importe quel nœud ou Pod dans le cluster, mais elle ne peut pas être atteinte de l'extérieur du cluster. Les valeurs pour `Endpoints` sont des adresses IP attribuées depuis votre VPC aux Pods qui font partie du service.

6. Visualisez les détails de l'un des Pods listés dans la sortie lorsque vous avez visualisé l'espace de noms dans une étape précédente. Si vous avez déployé une application Windows, remplacez *linux* par **windows** et remplacez *776d8f8fd8-78w66* par la valeur renvoyée pour l'un de vos Pods.

```
kubectl -n eks-sample-app describe pod eks-sample-linux-deployment-65b7669776-m6qxz
```

Sortie abrégée

Si vous avez déployé des ressources Windows, toutes les instances de *linux* dans la sortie suivante sont windows. Les autres *example values* peuvent être différents de votre sortie.

```
Name:         eks-sample-linux-deployment-65b7669776-m6qxz
Namespace:    eks-sample-app
Priority:     0
Node:         ip-192-168-45-132.us-west-2.compute.internal/192.168.45.132
[...]
IP:           192.168.63.93
IPs:
  IP:           192.168.63.93
Controlled By:  ReplicaSet/eks-sample-linux-deployment-65b7669776
[...]
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
[...]
Events:
  Type    Reason     Age    From
 Message
  ----    ------     ----   ----
 -------
  Normal  Scheduled  3m20s  default-scheduler
 Successfully assigned eks-sample-app/eks-sample-linux-deployment-65b7669776-m6qxz
 to ip-192-168-45-132.us-west-2.compute.internal
[...]
```

Dans la sortie précédente, la valeur de IP: est une IP unique attribuée au Pod à partir du bloc d'adresse CIDR attribué au sous-réseau dans lequel se trouve le nœud. Si vous préférez attribuer aux Pods des adresses IP à partir de différents blocs d'adresse CIDR, vous pouvez modifier le comportement par défaut. Pour plus d'informations, consultez Mise en réseau personnalisée pour les pods. Vous pouvez également voir que le planificateur Kubernetes a programmé le Pod sur le Node avec l'adresse IP *192.168.45.132*.

> **Tip**
>
> Plutôt que d'utiliser la ligne de commande, vous pouvez afficher de nombreux détails sur les Pods, les services, les déploiements et les autres ressources Kubernetes dans la AWS Management Console. Pour plus d'informations, consultez Afficher les ressources Kubernetes.

7. Exécutez un shell sur le Pod que vous avez décrit à l'étape précédente, en remplaçant *65b7669776-m6qxz* par l'ID de l'un de vos Pods.

   Linux

   ```
   kubectl exec -it eks-sample-linux-deployment-65b7669776-m6qxz -n eks-sample-app
    -- /bin/bash
   ```

   Windows

   ```
   kubectl exec -it eks-sample-windows-deployment-65b7669776-m6qxz -n eks-sample-
   app -- powershell.exe
   ```

8. À partir du shell du Pod, affichez la sortie du serveur web installé avec votre déploiement au cours d'une étape précédente. Vous devez uniquement spécifier le nom du service. Il est résolu en adresse IP du service par CoreDNS, qui est déployé avec un cluster Amazon EKS, par défaut.

   Linux

   ```
   curl eks-sample-linux-service
   ```

   L'exemple qui suit illustre un résultat.

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```

Windows

```
Invoke-WebRequest -uri eks-sample-windows-service/default.html -UseBasicParsing
```

L'exemple qui suit illustre un résultat.

```
StatusCode        : 200
StatusDescription : OK
Content           : < h t m l > < b o d y > < b r / > < b r / > < m a r q u e e
 > < H 1 > H e l l o
                    E K S ! ! ! < H 1 > < m a r q u e e > < / b o d y > < h t
 m l >
```

9. À partir du shell du Pod, affichez le serveur DNS pour le Pod.

Linux

```
cat /etc/resolv.conf
```

L'exemple qui suit illustre un résultat.

```
nameserver 10.100.0.10
search eks-sample-app.svc.cluster.local svc.cluster.local cluster.local us-
west-2.compute.internal
options ndots:5
```

Dans la sortie précédente, `10.100.0.10` est automatiquement affecté en tant que `nameserver` pour tous les Pods déployés dans le cluster.

Windows

```
Get-NetIPConfiguration
```

Sortie abrégée

```
InterfaceAlias       : vEthernet
[...]
IPv4Address          : 192.168.63.14
[...]
```

```
DNSServer            : 10.100.0.10
```

Dans la sortie précédente, `10.100.0.10` est automatiquement affecté en tant que serveur DNS pour tous les Pods déployés dans le cluster.

10. Déconnectez-vous du Pod en tapant `exit`.
11. Une fois que vous avez terminé avec l'application d'exemple, vous pouvez supprimer l'espace de noms, le service et le déploiement d'exemple avec la commande suivante.

```
kubectl delete namespace eks-sample-app
```

Étapes suivantes

Après avoir déployé l'exemple d'application, vous pouvez essayer certains des exercices suivants :

- the section called “Répartition de la charge des applications”
- the section called “Répartition de charge réseau”

Vertical Pod Autoscaler (VPA)

Le Vertical Pod Autoscaler de Kubernetes ajuste automatiquement les réservations de processeur et de mémoire pour vos Pods afin de vous aider à « dimensionner correctement » vos applications. Cet ajustement peut améliorer l'utilisation des ressources du cluster et libérer du processeur et de la mémoire pour d'autres Pods. Cette rubrique vous aide à déployer le VPA (Vertical Pod Autoscaler) sur votre cluster et à vérifier qu'il fonctionne.

Prérequis

- Vous disposez d'un cluster Amazon EKS existant. Si ce n'est pas le cas, consultez Démarrer avec Amazon EKS.
- Le serveur de métriques Kubernetes est installé. Pour plus d'informations, consultez Installation du serveur de métriques Kubernetes.
- Vous utilisez un client `kubectl` qui est configuré pour communiquer avec votre cluster Amazon EKS.
- OpenSSL `1.1.1` ou version ultérieure installé sur votre appareil.

Déploiement du VPA (Vertical Pod Autoscaler)

Dans cette section, vous déployez le Vertical Pod Autoscaler sur votre cluster.

Pour déployer le Vertical Pod Autoscaler

1. Ouvrez une fenêtre de terminal et accédez à un répertoire dans lequel vous souhaitez télécharger le code source du VPA (Vertical Pod Autoscaler).
2. Clonez le référentiel GitHub kubernetes/autoscaler.

   ```
   git clone https://github.com/kubernetes/autoscaler.git
   ```

3. Passez au répertoire `vertical-pod-autoscaler`.

   ```
   cd autoscaler/vertical-pod-autoscaler/
   ```

4. (Facultatif) Si vous avez déjà déployé une autre version du VPA (Vertical Pod Autoscaler), supprimez-la avec la commande suivante.

   ```
   ./hack/vpa-down.sh
   ```

5. Si vos nœuds n'ont pas d'accès Internet au registre de conteneurs `registry.k8s.io`, vous devez extraire les images suivantes et les transférer vers votre propre référentiel privé. Pour plus d'informations sur la manière d'extraire les images et de les transférer vers votre propre référentiel privé, consultez Copier une image de conteneur d'un référentiel vers un autre référentiel.

   ```
   registry.k8s.io/autoscaling/vpa-admission-controller:0.10.0
   registry.k8s.io/autoscaling/vpa-recommender:0.10.0
   registry.k8s.io/autoscaling/vpa-updater:0.10.0
   ```

   Si vous transférez les images vers un référentiel Amazon ECR privé, remplacez `registry.k8s.io` dans les manifestes par votre registre. Remplacez *`111122223333`* par votre ID de compte. Remplacez *`region-code`* par la Région AWS dans laquelle se trouve votre cluster. Les commandes suivantes supposent que vous avez nommé votre référentiel de la même manière que le référentiel du manifeste. Si vous avez nommé votre référentiel différemment, vous devrez également modifier son nom.

```
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/admission-controller-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/recommender-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/updater-deployment.yaml
```

6. Déployez le VPA (Vertical Pod Autoscaler) sur votre cluster avec la commande suivante.

```
./hack/vpa-up.sh
```

7. Vérifiez que les Pods Vertical Pod Autoscaler ont été créés avec succès.

```
kubectl get pods -n kube-system
```

L'exemple qui suit illustre un résultat.

```
NAME                                        READY   STATUS    RESTARTS   AGE
[...]
metrics-server-8459fc497-kfj8w              1/1     Running   0          83m
vpa-admission-controller-68c748777d-ppspd   1/1     Running   0          7s
vpa-recommender-6fc8c67d85-gljpl            1/1     Running   0          8s
vpa-updater-786b96955c-bgp9d                1/1     Running   0          8s
```

Test de l'installation de votre VPA (Vertical Pod Autoscaler)

Dans cette section, vous déployez un exemple d'application pour vérifier que le Vertical Pod Autoscaler fonctionne.

Pour tester votre installation Vertical Pod Autoscaler

1. Déployez l'exemple `hamster.yaml` Vertical Pod Autoscaler avec la commande suivante.

```
kubectl apply -f examples/hamster.yaml
```

2. Obtenez les Pods à partir de l'exemple d'application `hamster`.

```
kubectl get pods -l app=hamster
```

L'exemple qui suit illustre un résultat.

```
hamster-c7d89d6db-rglf5   1/1     Running   0          48s
hamster-c7d89d6db-znvz5   1/1     Running   0          48s
```

3. Décrivez l'un des Pods pour afficher sa réservation de `cpu` et de `memory`. Remplacez *`c7d89d6db-rglf5`* par l'un des ID renvoyés dans votre sortie de l'étape précédente.

```
kubectl describe pod hamster-c7d89d6db-rglf5
```

L'exemple qui suit illustre un résultat.

```
[...]
Containers:
  hamster:
    Container ID:  docker://
e76c2413fc720ac395c33b64588c82094fc8e5d590e373d5f818f3978f577e24
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:35:16 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        100m
      memory:     50Mi
[...]
```

Vous pouvez voir que le Pod d'origine réserve 100 millicpu de processeur et 50 mégaoctets de mémoire. Pour cet exemple d'application, 100 millicpu est inférieur à ce dont le Pod a besoin pour s'exécuter, il est donc limité par le processeur. Il réserve également beaucoup moins de mémoire qu'il n'en a besoin. Le déploiement du Vertical Pod Autoscaler `vpa-recommender`

analyse les Pods `hamster` pour voir si les exigences en termes de processeur et de mémoire sont appropriées. Si des ajustements sont nécessaires, le `vpa-updater` relance les Pods avec les valeurs mises à jour.

4. Attendez que le `vpa-updater` lance un nouveau Pod `hamster`. Cela devrait prendre une minute ou deux. Vous pouvez surveiller les Pods avec la commande suivante.

> **Note**
>
> Si vous n'êtes pas sûr qu'un nouveau Pod a été lancé, comparez les noms de Pod avec votre liste précédente. Lorsque le nouveau Pod est lancé, un nouveau nom de Pod s'affiche.

```
kubectl get --watch Pods -l app=hamster
```

5. Lorsqu'un nouveau Pod `hamster` est démarré, décrivez-le et affichez les réservations de processeur et de mémoire mises à jour.

```
kubectl describe pod hamster-c7d89d6db-jxgfv
```

L'exemple qui suit illustre un résultat.

```
[...]
Containers:
  hamster:
    Container ID:
 docker://2c3e7b6fb7ce0d8c86444334df654af6fb3fc88aad4c5d710eac3b1e7c58f7db
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:37:08 -0700
    Ready:          True
```

```
    Restart Count:  0
    Requests:
      cpu:        587m
      memory:     262144k
[...]
```

Dans la sortie précédente, vous pouvez voir que la réservation de l'`cpu` est passée à 587 millicpu, ce qui représente plus de cinq fois la valeur d'origine. La `memory` a augmenté jusqu'à 262 144 kilo-octets, soit environ 250 mébioctets ou cinq fois la valeur d'origine. Ce Pod était sous-approvisionné et le Vertical Pod Autoscaler a corrigé notre estimation avec une valeur beaucoup plus appropriée.

6. Décrivez la ressource `hamster-vpa` pour afficher la nouvelle recommandation.

```
kubectl describe vpa/hamster-vpa
```

L'exemple qui suit illustre un résultat.

```
Name:         hamster-vpa
Namespace:    default
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"autoscaling.k8s.io/
v1beta2","kind":"VerticalPodAutoscaler","metadata":{"annotations":
{},"name":"hamster-vpa","namespace":"d...
API Version:  autoscaling.k8s.io/v1beta2
Kind:         VerticalPodAutoscaler
Metadata:
  Creation Timestamp:  2019-09-27T18:22:51Z
  Generation:          23
  Resource Version:    14411
  Self Link:           /apis/autoscaling.k8s.io/v1beta2/namespaces/default/
verticalpodautoscalers/hamster-vpa
  UID:                 d0d85fb9-e153-11e9-ae53-0205785d75b0
Spec:
  Target Ref:
    API Version:  apps/v1
    Kind:         Deployment
    Name:         hamster
Status:
  Conditions:
    Last Transition Time:  2019-09-27T18:23:28Z
```

```
    Status:                True
    Type:                  RecommendationProvided
  Recommendation:
    Container Recommendations:
      Container Name:  hamster
      Lower Bound:
        Cpu:     550m
        Memory:  262144k
      Target:
        Cpu:     587m
        Memory:  262144k
      Uncapped Target:
        Cpu:     587m
        Memory:  262144k
      Upper Bound:
        Cpu:     21147m
        Memory:  387863636
Events:          <none>
```

7. Lorsque vous avez terminé de tester votre exemple d'application, supprimez-le avec la commande suivante.

```
kubectl delete -f examples/hamster.yaml
```

Horizontal Pod Autoscaler

Horizontal Pod Autoscaler de Kubernetes met automatiquement à l'échelle le nombre de Pods dans un déploiement, un contrôleur de réplication ou un ensemble de réplicas en fonction de l'utilisation du processeur de cette ressource. Cela peut permettre à vos applications de monter en puissance pour répondre à une demande accrue ou de baisser en puissance lorsque les ressources ne sont pas nécessaires, libérant ainsi vos nœuds pour d'autres applications. Lorsque vous définissez un pourcentage cible d'utilisation du processeur, Horizontal Pod Autoscaler met à l'échelle votre application pour essayer d'atteindre cet objectif.

Horizontal Pod Autoscaler est une ressource API standard dans Kubernetes qui nécessite simplement qu'une source de métriques (telle que le serveur de métriques Kubernetes) soit installée sur votre cluster Amazon EKS afin d'être opérationnel. Vous n'avez pas besoin de déployer ou d'installer Horizontal Pod Autoscaler sur votre cluster pour la mise à l'échelle de vos applications. Pour plus d'informations, consultez la section Horizontal Pod Autoscalerdans la documentation Kubernetes.

Utilisez cette rubrique pour préparer Horizontal Pod Autoscaler pour votre cluster Amazon EKS et pour vérifier qu'il fonctionne avec un exemple d'application.

> **Note**
>
> Cette rubrique est basée sur la démonstration Horizontal Pod autoscaler dans la documentation Kubernetes.

Prérequis

- Vous disposez d'un cluster Amazon EKS existant. Si ce n'est pas le cas, consultez Démarrer avec Amazon EKS.
- Le serveur de métriques Kubernetes est installé. Pour plus d'informations, consultez Installation du serveur de métriques Kubernetes.
- Vous utilisez un client `kubectl` qui est configuré pour communiquer avec votre cluster Amazon EKS.

Exécution d'une application de test du Horizontal Pod Autoscaler

Dans cette section, vous déployez un exemple d'application pour vérifier que Horizontal Pod Autoscale) fonctionne.

> **Note**
>
> Cet exemple est basé sur la démonstration de Horizontal Pod Autoscaler de la documentation Kubernetes.

Pour tester votre installation Horizontal Pod Autoscaler

1. Déployez une application simple de serveur web Apache avec la commande suivante.

   ```
   kubectl apply -f https://k8s.io/examples/application/php-apache.yaml
   ```

   Ce Pod de serveur web Apache est limité à 500 millicpu par processeur et fonctionne sur le port 80.
2. Créez une ressource Horizontal Pod Autoscaler pour le déploiement `php-apache`.

```
kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
```

Cette commande crée un autoscaler qui vise une utilisation de 50 % du processeur pour le déploiement, avec au minimum un Pod et au maximum dix Pods. Lorsque la charge moyenne du processeur est inférieure à 50 %, l'autoscaler tente de réduire le nombre de Pods dans le déploiement, jusqu'à un au munimum. Lorsque la charge est supérieure à 50 %, l'autoscaler essaie d'augmenter le nombre de Pods dans le déploiement, jusqu'à un maximum de dix. Pour plus d'informations, voir Comment HorizontalPodAutoscaler fonctionne un dans la Kubernetes documentation.

3. Décrivez l'autoscaler avec la commande suivante pour en afficher les détails.

```
kubectl get hpa
```

L'exemple qui suit illustre un résultat.

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   0%/50%    1         10        1          51s
```

Comme vous pouvez le voir, la charge actuelle CPU est de `0%`, car le serveur n'est pas encore chargé. Le nombre de Pod est déjà à sa limite inférieure (un), il ne peut donc pas être mis à l'échelle horizontale.

4. Créez une charge pour le serveur web en exécutant un conteneur.

```
kubectl run -i \
    --tty load-generator \
    --rm --image=busybox \
    --restart=Never \
    -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://php-apache; done"
```

5. Pour observer l'augmentation du déploiement, exécutez régulièrement la commande suivante dans un terminal distinct de celui dans lequel vous avez exécuté l'étape précédente.

```
kubectl get hpa php-apache
```

L'exemple qui suit illustre un résultat.

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
```

```
php-apache   Deployment/php-apache   250%/50%   1         10        5
 4m44s
```

L'augmentation du nombre de réplicas peut prendre plus d'une minute. Tant que le pourcentage de CPU réel est supérieur au pourcentage cible, le nombre de réplicas augmente jusqu'à 10. Dans ce cas, le pourcentage étant de `250%`, le nombre de `REPLICAS` continue d'augmenter.

> **Note**
>
> L'atteinte du nombre de réplicas maximum peut prendre un certain temps. Si, par exemple, seuls 6 réplicas sont nécessaires pour que la charge CPU reste égale ou inférieure à 50 %, la charge n'évoluera pas au-delà de 6 réplicas.

6. Arrêtez la charge. Dans la fenêtre de terminal dans laquelle vous générez la charge, arrêtez la charge en maintenant les touches `Ctrl+C` enfoncées. Vous pouvez voir les réplicas revenir à 1 en exécutant à nouveau la commande suivante dans le terminal dans lequel vous observez la mise à l'échelle horizontale.

```
kubectl get hpa
```

L'exemple qui suit illustre un résultat.

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   0%/50%    1         10        1          25m
```

> **Note**
>
> Le délai par défaut pour la réduction d'échelle est de cinq minutes. Par conséquent, un certain temps s'écoule avant de voir le nombre de réplicas atteindre à nouveau 1, même si le pourcentage de CPU actuel est de 0 %. Le délai est modifiable. Pour plus d'informations, consultez Horizontal Pod Autoscaler dans la documentation Kubernetes.

7. Lorsque vous avez terminé de tester votre exemple d'application, supprimez les ressources `php-apache`.

```
kubectl delete deployment.apps/php-apache service/php-apache
 horizontalpodautoscaler.autoscaling/php-apache
```

Répartition de charge réseau sur Amazon EKS

Le trafic réseau est équilibré en charge au niveau L4 du modèle OSI. Pour équilibrer la charge du trafic des applications surL7, vous déployez un Kubernetesingress, qui provisionne un AWS Application Load Balancer. Pour plus d'informations, consultez Répartition de la charge des applications sur Amazon EKS. Pour en savoir plus sur les différences entre les deux types d'équilibrage de charge, consultez les fonctionnalités d'Elastic Load Balancing sur le AWS site Web.

Lorsque vous créez un type Kubernetes Service ofLoadBalancer, le contrôleur d'équilibrage de charge du fournisseur de AWS cloud crée des équilibreurs de charge AWS classiques par défaut, mais peut également créer des équilibreurs de charge AWS réseau. Ce contrôleur ne recevra que des corrections de bugs critiques à l'avenir. Pour plus d'informations sur l'utilisation de l' AWS équilibreur de charge du fournisseur de AWS cloud, voir le contrôleur d'équilibrage de charge du fournisseur de cloud dans la Kubernetes documentation. Son utilisation n'est pas abordée dans cette rubrique.

Nous vous recommandons d'utiliser la version 2.7.2 ou ultérieure du AWS Load Balancer Controller en lieu et place du contrôleur de l'équilibreur de charge du fournisseur Cloud AWS . AWS Load Balancer Controllercrée des équilibreurs de charge AWS réseau, mais ne crée pas d'équilibreurs de charge AWS classiques. Le reste de cette rubrique traite de l'utilisation du AWS Load Balancer Controller.

Un AWS Network Load Balancer peut équilibrer la charge du trafic réseau à Pods déployer sur les cibles IP et les instances Amazon EC2 ou AWS Fargate sur des cibles IP. Pour plus d'informations, consultez AWS Load Balancer Controller sur GitHub.

Prérequis

Avant de pouvoir équilibrer la charge du trafic réseau à l'aide de l'AWS Load Balancer Controller, vous devez répondre aux exigences suivantes.

- Disposer d'un cluster. Si vous n'avez pas de cluster, consultez Démarrer avec Amazon EKS. Si vous devez mettre à jour la version d'un cluster existant, consultez Mise à jour d'une version Kubernetes de cluster Amazon EKS.
- Ayez le AWS Load Balancer Controller déployé sur votre cluster. Pour plus d'informations, consultez Qu'est-ce que AWS Load Balancer Controller ?. Nous recommandons la version 2.7.2 ou ultérieure.

- Au moins un sous-réseau. Si plusieurs sous-réseaux identifiés sont trouvés dans une zone de disponibilité, le contrôleur choisit le sous-réseau dont l'ID vient en premier dans l'ordre lexicographique. Le sous-réseau doit disposer au moins huit adresses IP disponibles.
- Si vous utilisez le contrôleur AWS Load Balancer Controller version `2.1.1` ou une version antérieure, les sous-réseaux doivent être labelisés comme suit. Si vous utilisez la version `2.1.2` ou une version ultérieure, cette identification est facultative. Vous souhaiterez peut-être baliser un sous-réseau si plusieurs clusters s'exécutent dans le même VPC, ou si AWS plusieurs services partagent des sous-réseaux dans un VPC, et si vous souhaitez mieux contrôler l'endroit où les équilibreurs de charge sont fournis pour chaque cluster. Si vous spécifiez explicitement les ID de sous-réseau en tant qu'annotation sur un objet de service, Kubernetes et l'AWS Load Balancer Controller utilisent directement ces sous-réseaux pour créer l'équilibreur de charge. L'identification de sous-réseau n'est pas nécessaire si vous choisissez d'utiliser cette méthode pour approvisionner les équilibreurs de charge, et vous pouvez ignorer les exigences suivantes d'identification de sous-réseau privé et public. Remplacez *`my-cluster`* par le nom de votre cluster.
  - Clé : `kubernetes.io/cluster/`*`my-cluster`*
  - Valeur : `shared` ou `owned`
- Vos sous-réseaux publics et privés doivent répondre aux exigences suivantes, sauf si vous spécifiez explicitement les ID de sous-réseau en tant qu'annotation sur un objet service ou d'entrée. Si vous allouez les équilibreurs de charge en spécifiant explicitement les ID de sous-réseau en tant qu'annotation sur un objet de service ou d'entrée, Kubernetes et l'AWS Load Balancer Controller utilisent directement ces sous-réseaux pour créer l'équilibreur de charge, et les balises suivantes ne sont pas nécessaires.
  - Sous-réseaux privés : doivent être étiquetés dans le format suivant. Cela permet au AWS Load Balancer Controller de savoir que les sous-réseaux peuvent être utilisés pour les équilibreurs de charge internes. Kubernetes Si vous utilisez `eksctl` un AWS AWS CloudFormation modèle Amazon EKS pour créer votre VPC après le 26 mars 2020, les sous-réseaux sont balisés de manière appropriée lors de leur création. Pour plus d'informations sur les modèles de VPC AWS AWS CloudFormation Amazon EKS, consultez Création d'un VPC pour votre cluster Amazon EKS.
    - Clé : `kubernetes.io/role/internal-elb`
    - Valeur : 1
  - Sous-réseau publics : doivent être étiquetés dans le format suivant. Ainsi, Kubernetes sait qu'il doit utiliser uniquement ces sous-réseaux pour les équilibreurs de charge externes au lieu de choisir un sous-réseau public dans chaque zone de disponibilité (en fonction de l'ordre

lexicographique des ID de sous-réseau). Si vous utilisez `eksctl` un AWS CloudFormation modèle Amazon EKS pour créer votre VPC après le 26 mars 2020, les sous-réseaux sont balisés de manière appropriée lors de leur création. Pour plus d'informations sur les modèles de AWS CloudFormation VPC Amazon EKS, consultez. Création d'un VPC pour votre cluster Amazon EKS

- Clé : `kubernetes.io/role/elb`
- Valeur : `1`

Si les balises de rôle de sous-réseau ne sont pas explicitement ajoutées, le contrôleur de service Kubernetes examine la table de routage des sous-réseaux VPC de votre cluster pour déterminer si le sous-réseau est privé ou public. Nous vous recommandons de ne pas vous fier à ce comportement et d'ajouter explicitement les identifications de rôle privées ou publiques. L'AWS Load Balancer Controller n'examine pas les tables de routage et exige que les identifications privées et publiques soient présentes pour que la découverte automatique fonctionne.

Considérations

- La configuration de votre équilibreur de charge est contrôlée par des annotations qui sont ajoutées au manifeste de votre service. Les annotations de service sont différentes lorsque vous AWS Load Balancer Controller utilisez le contrôleur d'équilibrage de charge du fournisseur de AWS cloud. Veillez à consulter les annotations de l'AWS Load Balancer Controller avant de déployer les services.
- Lorsque vous utilisez le Amazon VPC CNI plugin for Kubernetes, l'AWS Load Balancer Controller peut équilibrer la charge vers des cibles IP ou d'instance Amazon EC2 et des cibles IP Fargate. Si vous utilisez d'autres plugins CNI compatibles, le contrôleur peut uniquement équilibrer la charge vers des cibles d'instance. Pour plus d'informations sur les types de cibles de Network Load Balancer, veuillez consulter la section Type de cible dans le guide de l'utilisateur de Network Load Balancer
- Si vous souhaitez ajouter des balises à l'équilibreur de charge lors de sa création ou après, ajoutez l'annotation suivante dans votre spécification de service. Pour plus d'informations, consultez Identifications de ressources AWS dans la documentation du contrôleur de l'AWS Load Balancer Controller.

```
service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags
```

- Vous pouvez affecter des adresses IP Elastic au Network Load Balancer en ajoutant l'annotation suivante. Remplacez les *example values* par les `Allocation IDs` de vos adresses IP élastiques. Le nombre de `Allocation IDs` doit correspondre au nombre de sous-réseaux utilisés pour l'équilibreur de charge. Pour de plus amples informations, veuillez consulter la documentation AWS Load Balancer Controller.

```
service.beta.kubernetes.io/aws-load-balancer-eip-allocations:
 eipalloc-xxxxxxxxxxxxxxxxx,eipalloc-yyyyyyyyyyyyyyyyy
```

- Amazon EKS ajoute une règle entrante au groupe de sécurité du nœud pour le trafic client et une règle pour chaque sous-réseau d'équilibreur de charge dans le VPC pour les surveillances de l'état pour chaque Network Load Balancer que vous créez. Le déploiement d'un service de type `LoadBalancer` peut échouer si Amazon EKS tente de créer des règles qui dépassent le quota du nombre maximal de règles autorisées pour un groupe de sécurité. Pour plus d'informations, consultez Groupes de sécurité dans Quotas de VPC Amazon dans le guide de l'utilisateur d'Amazon VPC. Tenez compte des options suivantes pour réduire les possibilités de dépassement du nombre maximum de règles pour un groupe de sécurité :
  - Demandez une augmentation dans vos règles par quota de groupe de sécurité. Pour plus d'informations, consultez Demande d'une augmentation de quota dans le Guide de l'utilisateur de Service Quotas.
  - Utilisez des cibles IP plutôt que des cibles d'instance. Avec les cibles IP, vous pouvez partager des règles pour les mêmes ports cibles. Vous pouvez spécifier manuellement les sous-réseaux des équilibreurs de charge avec une annotation. Pour plus d'informations, consultez la rubrique Annotations sur GitHub.
  - Utilisez une entrée (ingress), au lieu d'un service de type `LoadBalancer`, pour envoyer le trafic vers votre service. L' AWS Application Load Balancer nécessite moins de règles que les Network Load Balancer. Vous pouvez partager un ALB sur plusieurs entrées. Pour plus d'informations, consultez Répartition de la charge des applications sur Amazon EKS. Vous ne pouvez pas partager un Network Load Balancer entre plusieurs services.
  - Déployez vos clusters dans plusieurs comptes.
- Si vos Pods fonctionnent sur Windows dans un cluster Amazon EKS, un seul service avec un équilibreur de charge peut prendre en charge jusqu'à 1024 Pods de backend. Chaque Pod a sa propre adresse IP.
- Nous recommandons de ne créer de nouveaux Network Load Balancers qu'avec l'AWS Load Balancer Controller. Toute tentative de remplacement des équilibreurs de charge réseau existants créés à l'aide du contrôleur d'équilibrage de charge du fournisseur de AWS cloud peut entraîner la

création de plusieurs équilibreurs de charge réseau susceptibles de provoquer des interruptions de service des applications.

Créer un équilibreur de charge de réseau

Vous pouvez créer un équilibreur de charge de réseau avec des cibles IP ou d'instance.

IP targets

Vous pouvez utiliser des cibles IP avec des Pods déployés sur des nœuds Amazon EC2 ou Fargate. Votre service Kubernetes doit être créé en tant que type `LoadBalancer`. Pour plus d'informations, consultez la section Type LoadBalancer dans la Kubernetes documentation.

Pour créer un équilibreur de charge qui utilise des cibles IP, ajoutez l'annotation suivante à un manifeste du service et déployez votre service. La `external` valeur de `aws-load-balancer-type` est ce qui pousse le AWS Load Balancer Controller contrôleur d'équilibrage de charge du fournisseur de AWS cloud à créer le Network Load Balancer, plutôt que le contrôleur du fournisseur de cloud. Vous pouvez visualiser un exemple de manifeste de service avec les annotations.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

> **Note**
>
> Si vous effectuez un équilibrage de charge vers des Pods `IPv6`, ajoutez l'annotation suivante. Vous ne pouvez effectuer un équilibrage de charge sur `IPv6` que vers des cibles IP, pas vers des cibles d'instance. Sans cette annotation, l'équilibrage de charge passe par `IPv4`.
>
> ```
> service.beta.kubernetes.io/aws-load-balancer-ip-address-type: dualstack
> ```

Les Network Load Balancer sont créés avec le `internal` `aws-load-balancer-scheme`, par défaut. Vous pouvez lancer des équilibreurs de charge réseau dans n'importe quel sous-réseau du VPC de votre cluster, y compris dans les sous-réseaux non spécifiés lors de la création de votre cluster.

Kubernetes examine la table de routage de vos sous-réseaux afin d'identifier s'ils sont publics ou privés. Les sous-réseaux publics dispose d'un acheminement direct vers Internet en utilisant une passerelle Internet. Ce n'est pas le cas des sous-réseaux privés.

Si vous souhaitez créer un Network Load Balancer dans un sous-réseau public pour équilibrer la charge vers les nœuds Amazon EC2 (Fargate ne peut être que privé), spécifiez `internet-facing` avec l'annotation suivante :

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Note**
>
> Pour des raisons de rétrocompatibilité, l'annotation `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"` est toujours prise en charge. Cependant, nous vous recommandons d'utiliser les annotations précédentes pour les nouveaux équilibreurs de charge au lieu de `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`.

> **Important**
>
> Ne modifiez pas les annotations après la création de votre service. Si vous devez le modifier, supprimez l'objet de service et créez-le à nouveau avec la valeur souhaitée pour cette annotation.

Instance targets

Le contrôleur d'équilibrage de charge du fournisseur de AWS cloud crée des équilibreurs de charge réseau avec des cibles d'instance uniquement. Les versions `2.2.0` et ultérieures de l' AWS Load Balancer Controller créent également des Network Load Balancers avec des cibles d'instance. Nous vous recommandons de l'utiliser, plutôt que le contrôleur d'équilibrage de charge du fournisseur de AWS cloud, pour créer de nouveaux équilibreurs de charge réseau. Vous pouvez utiliser des cibles d'instance Network Load Balancer avec des Pods déployés sur des nœuds Amazon EC2, mais pas sur Fargate. Pour équilibrer la charge du trafic réseau entre les Pods déployés sur Fargate, vous devez utiliser des cibles IP.

Pour déployer un Network Load Balancer sur un sous-réseau privé, votre spécification de service doit comporter les annotations suivantes. Vous pouvez visualiser un exemple de manifeste de service avec les annotations. La `external` valeur de `aws-load-balancer-type` est ce qui pousse le AWS Load Balancer Controller, plutôt que le contrôleur d'équilibrage de charge du fournisseur de AWS cloud, à créer le Network Load Balancer.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "instance"
```

Les Network Load Balancer sont créés avec le `internal` `aws-load-balancer-scheme`, par défaut. Pour les Network Load Balancer internes, votre cluster Amazon EKS doit être configuré pour utiliser au moins un sous-réseau privé dans votre VPC. Kubernetes examine la table de routage de vos sous-réseaux pour identifier s'ils sont publics ou privés. Les sous-réseaux publics dispose d'un acheminement direct vers Internet en utilisant une passerelle Internet. Ce n'est pas le cas des sous-réseaux privés.

Si vous souhaitez créer un Network Load Balancer dans un sous-réseau public pour équilibrer la charge vers les nœuds Amazon EC2, spécifiez `internet-facing` avec l'annotation suivante :

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Important**
>
> Ne modifiez pas les annotations après la création de votre service. Si vous devez le modifier, supprimez l'objet de service et créez-le à nouveau avec la valeur souhaitée pour cette annotation.

(Facultatif) Déployer un exemple d'application

Prérequis

- Au moins un sous-réseau public ou privé dans votre VPC de cluster.
- Ayez le AWS Load Balancer Controller déployé sur votre cluster. Pour plus d'informations, consultez Qu'est-ce que AWS Load Balancer Controller ?. Nous recommandons la version `2.7.2` ou ultérieure.

Pour déployer un exemple d'application

1. Si vous déployez vers Fargate, assurez-vous que vous disposez d'un sous-réseau privé disponible dans votre VPC et créez un profil Fargate. Si vous ne déployez pas sur Fargate, ignorez cette étape. Vous pouvez créer le profil en exécutant la commande suivante ou dans la AWS Management Console en utilisant les mêmes valeurs pour `name` et `namespace` que dans la commande. Remplacez les *`example values`* par vos propres valeurs.

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name nlb-sample-app \
       --namespace nlb-sample-app
   ```

2. Déployez un exemple d'application.

   a. Créez un espace de nom pour l'application.

      ```
      kubectl create namespace nlb-sample-app
      ```

   b. Enregistrez le contenu suivant dans un fichier nommé *`sample-deployment`*`.yaml` sur votre ordinateur.

      ```
      apiVersion: apps/v1
      kind: Deployment
      metadata:
        name: nlb-sample-app
        namespace: nlb-sample-app
      spec:
        replicas: 3
        selector:
          matchLabels:
            app: nginx
        template:
          metadata:
            labels:
              app: nginx
          spec:
            containers:
              - name: nginx
                image: public.ecr.aws/nginx/nginx:1.23
                ports:
      ```

```
            - name: tcp
              containerPort: 80
```

c. Appliquez le fichier manifeste à votre cluster.

```
kubectl apply -f sample-deployment.yaml
```

3. Créez un service avec un Network Load Balancer interne qui équilibre la charge vers les cibles IP.

a. Enregistrez le contenu suivant dans un fichier nommé *sample-service*.yaml sur votre ordinateur. Si vous déployez vers des nœuds Fargate, supprimez la ligne `service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing`.

```
apiVersion: v1
kind: Service
metadata:
  name: nlb-sample-service
  namespace: nlb-sample-app
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

b. Appliquez le fichier manifeste à votre cluster.

```
kubectl apply -f sample-service.yaml
```

4. Vérifiez que le service a été déployé.

```
kubectl get svc nlb-sample-service -n nlb-sample-app
```

L'exemple qui suit illustre un résultat.

```
NAME             TYPE           CLUSTER-IP         EXTERNAL-IP
                                                  PORT(S)        AGE
sample-service   LoadBalancer   10.100.240.137
 k8s-nlbsampl-nlbsampl-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
  80:32400/TCP   16h
```

> **Note**
>
> Les valeurs pour *10.100.240.137* et *xxxxxxxxxx* - *xxxxxxxxxxxxxxxx* seront différentes de celles de l'exemple de sortie (elles seront propres à votre équilibreur de charge) et peut être différent pour vous, selon le cluster dans lequel se trouve votre cluster. Région AWS

5. Ouvrez Amazon EC2. AWS Management Console Sélectionnez Target Groups (Groupes cibles) (sous Load Balancing [Équilibrage de charge]) dans le panneau de navigation de gauche. Dans la colonne Name (Nom), sélectionnez le nom du groupe cible où la valeur de la colonne Load balancer (Équilibreur de charge) correspond à une partie du nom dans la colonne `EXTERNAL-IP` de la sortie à l'étape précédente. Par exemple, vous sélectionneriez le groupe cible nommé `k8s-default-samplese-`*xxxxxxxxxx* si votre sortie était la même que celle de l'étape précédente. Le Type de cible est `IP`, car cela a été spécifié dans l'exemple de manifeste de service.
6. Sélectionnez votre Target group (Groupe cible), puis cliquez sur l'onglet Targets (Cibles). Sous Cibles enregistrées, vous devez voir trois adresses IP des trois réplicas déployés à une étape précédente. Attendez que le statut de toutes les cibles soit sain avant de continuer. L'affichage du statut `healthy` pour toutes les cibles peut prendre plusieurs minutes. Les cibles peuvent avoir l'état `unhealthy` avant qu'il soit remplacé par `healthy`.
7. Envoyer le trafic vers le service en remplaçant *xxxxxxxxxx-xxxxxxxxxxxxxxxx* et *us-west-2* par les valeurs renvoyées dans la sortie de l'étape précédente pour `EXTERNAL-IP`. Si vous avez déployé sur un sous-réseau privé, vous devez afficher la page à partir d'un appareil de votre VPC, tel qu'un hôte bastion. Pour plus d'informations, consultez la section Hôtes bastions Linux sur AWS.

```
curl k8s-default-samplese-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
```

L'exemple qui suit illustre un résultat.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

8. Lorsque vous avez terminé avec le déploiement, le service et l'espace de noms de l'exemple, supprimez-les.

```
kubectl delete namespace nlb-sample-app
```

Répartition de la charge des applications sur Amazon EKS

Lorsque vous créez un Kubernetes`ingress`, un AWS Application Load Balancer (ALB) est configuré pour équilibrer la charge du trafic des applications. Pour en savoir plus, consultez Qu'est-ce qu'un Application Load Balancer ? dans le Guide de l'utilisateur Application Load Balancers et Ingress dans la documentation Kubernetes. Les ALB peuvent être utilisés avec des Pods qui sont déployés sur des nœuds ou sur AWS Fargate. Vous pouvez déployer un ALB sur des sous-réseaux publics ou privés.

Le trafic des applications est équilibré au `L7` du modèle OSI. Pour équilibrer la charge du trafic réseau au `L4`, déployez un Kubernetes `service`de type `LoadBalancer`. Ce type fournit un AWS Network Load Balancer. Pour plus d'informations, consultez Répartition de charge réseau sur Amazon EKS. Pour en savoir plus sur les différences entre les deux types d'équilibreurs de charge, consultez Caractéristiques Elastic Load Balancing sur le site AWS .

Prérequis

Pour pouvoir équilibrer la charge du trafic d'une application, vous devez remplir les conditions suivantes.

- Disposer d'un cluster. Si vous n'avez pas de cluster, consultez Démarrer avec Amazon EKS. Si vous devez mettre à jour la version d'un cluster existant, consultez Mise à jour d'une version Kubernetes de cluster Amazon EKS.
- Ayez le AWS Load Balancer Controller déployé sur votre cluster. Pour plus d'informations, consultez Qu'est-ce que AWS Load Balancer Controller ?. Nous recommandons la version `2.7.2` ou ultérieure.

- Au moins deux sous-réseaux dans des zones de disponibilité différentes. Le AWS Load Balancer Controller choisit un sous-réseau dans chaque zone de disponibilité. Lorsque plusieurs sous-réseaux étiquetés sont trouvés dans une zone de disponibilité, le contrôleur choisit le sous-réseau dont l'ID vient en premier par ordre lexicographique. Chaque sous-réseau doit disposer au moins huit adresses IP disponibles.

  Si vous utilisez plusieurs groupes de sécurité attachés au composant master, un seul groupe de sécurité doit être étiqueté comme suit. Remplacez *my-cluster* par le nom de votre cluster.

  - Clé : `kubernetes.io/cluster/`*my-cluster*
  - Valeur : `shared` ou `owned`

- Si vous utilisez AWS Load Balancer Controller version `2.1.1` ou une version antérieure, les sous-réseaux doivent être étiquetés dans le format suivant. Si vous utilisez la version `2.1.2` ou une version ultérieure, l'étiquetage est facultatif. Cependant, nous vous recommandons d'étiqueter un sous-réseau dans les cas suivants. Plusieurs clusters s'exécutent dans le même VPC ou plusieurs AWS services partagent des sous-réseaux au sein d'un VPC. Sinon, vous souhaitez avoir plus de contrôle sur l'endroit où les équilibreurs de charge sont alloués pour chaque cluster. Remplacez *my-cluster* par le nom de votre cluster.

  - Clé : `kubernetes.io/cluster/`*my-cluster*
  - Valeur : `shared` ou `owned`

- Vos sous-réseaux publics et privés doivent répondre aux critères suivants : Ceci s'applique si vous ne spécifiez pas explicitement les ID de sous-réseau en tant qu'annotation sur un objet service ou d'entrée. Supposons que vous allouiez des équilibreurs de charge en spécifiant explicitement les ID de sous-réseau en tant qu'annotation sur un objet service ou d'entrée. Dans ce cas, Kubernetes et le contrôleur de l'équilibreur de charge AWS utilisent directement ces sous-réseaux pour créer l'équilibreur de charge et les balises suivantes ne sont pas nécessaires.

  - Sous-réseaux privés : doivent être étiquetés dans le format suivant. Cela permet au contrôleur d'équilibreur de AWS charge de savoir que les sous-réseaux peuvent être utilisés pour les équilibreurs de charge internes. Kubernetes Si vous utilisez `eksctl` un AWS CloudFormation modèle Amazon EKS pour créer votre VPC après le 26 mars 2020, les sous-réseaux sont balisés de manière appropriée lors de leur création. Pour plus d'informations sur les modèles de AWS CloudFormation VPC Amazon EKS, consultez. Création d'un VPC pour votre cluster Amazon EKS

    - Clé : `kubernetes.io/role/internal-elb`
    - Valeur : `1`

- Sous-réseau publics : doivent être étiquetés dans le format suivant. Ainsi, Kubernetes sait qu'il doit utiliser uniquement les sous-réseaux qui ont été spécifiés pour les équilibreurs de charge externes. De cette façon, Kubernetes ne choisit pas un sous-réseau public dans chaque zone de disponibilité (lexicographiquement basé sur son ID de sous-réseau). Si vous utilisez `eksctl` un AWS CloudFormation modèle Amazon EKS pour créer votre VPC après le 26 mars 2020, les sous-réseaux sont balisés de manière appropriée lors de leur création. Pour plus d'informations sur les modèles de AWS CloudFormation VPC Amazon EKS, consultez. Création d'un VPC pour votre cluster Amazon EKS
  - Clé : `kubernetes.io/role/elb`
  - Valeur : `1`

Si les balises de rôle de sous-réseau ne sont pas explicitement ajoutées, le contrôleur de service Kubernetes examine la table de routage des sous-réseaux de votre VPC de cluster. Cela permet de déterminer si le sous-réseau est privé ou public. Nous vous recommandons de ne pas vous fier à ce comportement. Ajoutez plutôt explicitement les identifications de rôle privées ou publiques. Le AWS Load Balancer Controller n'examine pas les tables de routage. Il est également nécessaire que les identifications privées et publiques soient présentes pour que la découverte automatique fonctionne.

Considérations

- Le AWS Load Balancer Controller crée des ALB et les AWS ressources de support nécessaires chaque fois qu'une ressource d'Kubernetesentrée est créée sur le cluster avec l'annotation. `kubernetes.io/ingress.class: alb` La ressource d'entrée configure l'ALB pour acheminer le trafic HTTP ou HTTPS vers les différents Pods du cluster. Pour que vos objets d'entrée utilisent le AWS Load Balancer Controller, ajoutez l'annotation suivante à votre spécification d'entrée Kubernetes. Pour plus d'informations, consultez Spécification d'entrée sur GitHub.

```
annotations:
    kubernetes.io/ingress.class: alb
```

> **Note**
>
> Si vous effectuez un équilibrage de charge vers des `IPv6` Pods, ajoutez l'annotation suivante à votre spécification d'entrée. Vous ne pouvez effectuer un équilibrage de charge

sur IPv6 que vers des cibles IP, pas vers des cibles d'instance. Sans cette annotation, l'équilibrage de charge passe par IPv4.

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

- Le AWS Load Balancer Controller prend en charge les modes de trafic suivants :
  - Instance : enregistre les nœuds de votre cluster comme cibles de l'ALB. Le trafic atteignant l'ALB est acheminé vers le NodePort de votre service, puis transmis par proxy à vos Pods. Il s'agit du mode de trafic par défaut. Vous pouvez également le spécifier explicitement avec l'annotation `alb.ingress.kubernetes.io/target-type: instance`.

    > **Note**
    >
    > Votre Kubernetes service doit spécifier le type NodePort ou « LoadBalancer » pour utiliser ce mode de trafic.

  - IP : enregistre les Pods comme cibles pour l'ALB. Le trafic atteignant l'ALB est directement acheminé vers des Pods de votre service. Vous devez spécifier l'annotation `alb.ingress.kubernetes.io/target-type: ip` pour pouvoir utiliser ce mode de trafic. Le type de cible IP est requis lorsque les Pods cibles fonctionnent sur Fargate.
- Pour identifier les ALB créés par le contrôleur, ajoutez l'annotation suivante au contrôleur : `alb.ingress.kubernetes.io/tags`. Pour obtenir la liste des annotations disponibles prises en charge par le AWS Load Balancer Controller, consultez Annotations d'entrée sur GitHub.
- La mise à niveau ou la rétrogradation de la version du contrôleur ALB peut introduire des changements de rupture pour les fonctionnalités qui en dépendent. Pour plus d'informations sur les changements importants introduits dans chaque version, consultez les notes de mise à jour du contrôleur ALB sur GitHub.

Pour partager un Application Load Balancer sur plusieurs ressources d'entrée à l'aide de **`IngressGroups`**

Pour joindre une entrée à un groupe, ajoutez l'annotation suivante à une spécification de ressource d'entrée Kubernetes.

```
alb.ingress.kubernetes.io/group.name: my-group
```

Le nom du groupe doit :

- Contenir 63 caractères maximum.
- Contenir des minuscules, des chiffres, des `-` et des `.`.
- Commencer et se terminer par un chiffre ou une lettre.

Le contrôleur fusionne automatiquement les règles d'entrée pour toutes les entrées du même groupe d'entrées. Il les prend en charge avec un seul ALB. La plupart des annotations qui sont définies sur une ressource d'entrée ne s'appliquent qu'aux chemins définis par cette ressource. Par défaut, les ressources d'entrée n'appartiennent à aucun groupe d'entrée.

> **Warning**
>
> Risques potentiels de sécurité : ne spécifiez un groupe d'entrée pour une entrée que lorsque tous les utilisateurs Kubernetes qui ont l'autorisation RBAC de créer ou de modifier des ressources d'entrée se trouvent dans la même limite d'approbation. Si vous ajoutez l'annotation avec un nom de groupe, d'autres utilisateurs Kubernetes peuvent créer ou modifier leurs entrées pour appartenir au même groupe d'entrée. Cela peut entraîner un comportement indésirable, comme l'écrasement des règles existantes par des règles de priorité supérieure.

Vous pouvez ajouter le numéro d'ordre de votre ressource d'entrée.

```
alb.ingress.kubernetes.io/group.order: '10'
```

Le numéro peut être 1-1000. Le numéro le plus bas de toutes les ressources d'entrée d'un même groupe d'entrée est évalué en premier. Toutes les ressources d'entrée sans cette annotation sont évaluées avec la valeur zéro. Les règles dupliquées avec un numéro supérieur peuvent écraser les règles avec un numéro inférieur. Par défaut, l'ordre des règles entre les ressources d'entrée d'un même groupe d'entrée est déterminé de manière lexicographique sur la base de l'espace de noms et du nom.

> **Important**
>
> Assurez-vous que chaque ressource d'un groupe d'entrée dispose d'un numéro de priorité unique. Vous ne pouvez pas avoir des numéros d'ordre en double dans les ressources d'entrée.

(Facultatif) Déployer un exemple d'application

Prérequis

- Au moins un sous-réseau public ou privé dans votre VPC de cluster.
- Ayez le AWS Load Balancer Controller déployé sur votre cluster. Pour plus d'informations, consultez Qu'est-ce que AWS Load Balancer Controller ?. Nous recommandons la version `2.7.2` ou ultérieure.

Pour déployer un exemple d'application

Vous pouvez exécuter l'exemple d'application dans un cluster comportant des nœuds Amazon EC2, des Pods Fargate ou les deux.

1. Si vous ne déployez pas sur Fargate, ignorez cette étape. Si vous déployez sur Fargate, créez un profil Fargate. Vous pouvez créer le profil en exécutant la commande suivante ou dans la AWS Management Console en utilisant les mêmes valeurs pour `name` et `namespace` que dans la commande. Remplacez les *`example values`* par vos propres valeurs.

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name alb-sample-app \
       --namespace game-2048
   ```

2. Déployez le jeu 2048 en tant qu'exemple d'application pour vérifier que l'objet d'entrée AWS Load Balancer Controller crée un AWS ALB. Effectuez les étapes pour le type de sous-réseau sur lequel vous déployez.

   a. Si vous effectuez un déploiement vers des Pods dans un cluster que vous avez créé avec la famille `IPv6`, passez à l'étape suivante.

- Public

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
```

- Privé

1. Téléchargez le manifeste.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
```

2. Modifiez le fichier et trouvez la ligne qui contient `alb.ingress.kubernetes.io/scheme: internet-facing`.
3. Remplacez *`internet-facing`* par **`internal`** et enregistrez le fichier.
4. Appliquez le manifeste à votre cluster.

```
kubectl apply -f 2048_full.yaml
```

b. Si vous effectuez un déploiement vers des Pods dans un cluster que vous avez créé avec la famille `IPv6`, effectuez les étapes suivantes.

1. Téléchargez le manifeste.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
```

2. Ouvrez le fichier dans un éditeur et ajoutez la ligne suivante aux annotations de la spécification d'entrée.

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

3. Si vous effectuez un équilibrage de charge vers des Pods internes, plutôt que des Pods orientés vers Internet, modifiez la ligne qui indique `alb.ingress.kubernetes.io/scheme: internet-facing` à `alb.ingress.kubernetes.io/scheme: internal`
4. Enregistrez le fichier.
5. Appliquez le manifeste à votre cluster.

```
kubectl apply -f 2048_full.yaml
```

3. Après quelques minutes, vérifiez que la ressource d'entrée a été créée en utilisant la commande suivante.

```
kubectl get ingress/ingress-2048 -n game-2048
```

L'exemple qui suit illustre un résultat.

```
NAME           CLASS    HOSTS   ADDRESS
                       PORTS   AGE
ingress-2048   <none>   *       k8s-game2048-ingress2-xxxxxxxxxx-yyyyyyyyyy.region-
code.elb.amazonaws.com   80      2m32s
```

> **Note**
>
> Si vous avez créé l'équilibreur de charge dans un sous-réseau privé, la valeur sous `ADDRESS` dans la sortie précédente est préfacée avec `internal-`.
> Si votre ressource d'entrée n'a pas été créé après plusieurs minutes, exécutez la commande suivante pour afficher les journaux du AWS Load Balancer Controller.
> Ces journaux peuvent contenir des messages d'erreur que vous pouvez utiliser pour diagnostiquer les problèmes de votre déploiement.
>
> ```
> kubectl logs -f -n kube-system -l app.kubernetes.io/instance=aws-load-
> balancer-controller
> ```

4. Si vous l'avez déployé dans un sous-réseau public, ouvrez un navigateur et naviguez vers l'URL `ADDRESS` de la sortie de commande précédente pour voir l'exemple d'application. Si vous ne voyez rien, actualisez votre navigateur et réessayez. Si vous avez déployé sur un sous-réseau privé, vous devez afficher la page à partir d'un appareil de votre VPC, tel qu'un hôte bastion. Pour plus d'informations, consultez la section Hôtes bastions Linux sur AWS.
5. Lorsque vous avez terminé de tester l'exemple d'application, supprimez-le avec les commandes suivantes.
   - Si vous avez appliqué le manifeste plutôt que d'appliquer une copie que vous avez téléchargée, utilisez la commande suivante.

```
kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
```

- Si vous avez téléchargé et modifié le manifeste, utilisez la commande suivante.

```
kubectl delete -f 2048_full.yaml
```

Restriction des adresses IP externes pouvant être affectées à des services

Les services Kubernetes peuvent être atteints depuis l'intérieur d'un cluster via :

- Une adresse IP de cluster est affectée automatiquement par Kubernetes
- Toute adresse IP que vous spécifiez pour la propriété `externalIPs` dans une spécification de service. Les adresses IP externes ne sont pas gérées par Kubernetes et relèvent de la responsabilité de l'administrateur du cluster. Les adresses IP externes spécifiées avec `externalIPs` sont différentes de l'adresse IP externe affectée à un service de type `LoadBalancer` par un fournisseur de cloud.

Pour en savoir plus sur les services Kubernetes, consultez Service dans la documentation Kubernetes. Vous pouvez restreindre les adresses IP qui peuvent être spécifiées pour `externalIPs` dans une spécification de service.

Pour restreindre les adresses IP qui peuvent être spécifiées pour **`externalIPs`** dans une spécification de service

1. Déployez `cert-manager` pour gérer les certificats webhook. Pour de plus amples informations, veuillez consulter la documentation `cert-manager`.

```
kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/cert-manager.yaml
```

2. Vérifiez que les Pods `cert-manager` sont en cours d'exécution.

```
kubectl get pods -n cert-manager
```

L'exemple qui suit illustre un résultat.

```
NAME                                       READY   STATUS    RESTARTS   AGE
cert-manager-58c8844bb8-nlx7q              1/1     Running   0          15s
cert-manager-cainjector-745768f6ff-696h5   1/1     Running   0          15s
cert-manager-webhook-67cc76975b-4v4nk      1/1     Running   0          14s
```

3. Vérifiez vos services existants pour vous assurer qu'aucun d'entre eux ne possède des adresses IP externes qui leur sont affectées et qui ne sont pas contenues dans le bloc d'adresse CIDR auquel vous souhaitez limiter les adresses.

```
kubectl get services -A
```

L'exemple qui suit illustre un résultat.

```
NAMESPACE                       NAME                                      TYPE
   CLUSTER-IP       EXTERNAL-IP     PORT(S)         AGE
cert-manager                    cert-manager                              ClusterIP
   10.100.102.137   <none>          9402/TCP        20m
cert-manager                    cert-manager-webhook                      ClusterIP
   10.100.6.136     <none>          443/TCP         20m
default                         kubernetes                                ClusterIP
   10.100.0.1       <none>          443/TCP         2d1h
externalip-validation-system    externalip-validation-webhook-service     ClusterIP
   10.100.234.179   <none>          443/TCP         16s
kube-system                     kube-dns                                  ClusterIP
   10.100.0.10      <none>          53/UDP,53/TCP   2d1h
my-namespace                    my-service                                ClusterIP
   10.100.128.10    192.168.1.1     80/TCP          149m
```

Si certaines des valeurs sont des adresses IP qui ne se trouvent pas dans le bloc auquel vous souhaitez restreindre l'accès, vous devez modifier les adresses pour qu'elles se trouvent dans le bloc et redéployer les services. Par exemple, le service `my-service` dans la sortie précédente a une adresse IP externe affectée qui ne se trouve pas dans l'exemple de bloc d'adresse CIDR à l'étape 5.

4. Téléchargez le manifeste du webhook IP externe. Vous pouvez également afficher le code source pour le webhook sur GitHub.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/externalip-webhook.yaml
```

5. Spécifiez des blocs d'adresse CIDR. Ouvrez le fichier téléchargé dans l'éditeur et supprimez le # au début des lignes suivantes.

```
#args:
#- --allowed-external-ip-cidrs=10.0.0.0/8
```

Remplacez `10.0.0.0/8` par votre propre bloc d'adresse CIDR. Vous pouvez spécifier autant de blocs que vous le souhaitez. Si vous spécifiez plusieurs blocs, ajoutez une virgule entre les blocs.

6. Si votre cluster n'est pas dans la Région AWS `us-west-2`, rmplacez `us-west-2`, `602401143452` et `amazonaws.com` dans le fichier avec les commandes suivantes. Avant d'exécuter les commandes, remplacez *`region-code`* et *`111122223333`* par la valeur de votre Région AWS dans la liste figurant dans les Registres d'images de conteneur Amazon.

```
sed -i.bak -e 's|602401143452|111122223333|' externalip-webhook.yaml
sed -i.bak -e 's|us-west-2|region-code|' externalip-webhook.yaml
sed -i.bak -e 's|amazonaws.com||' externalip-webhook.yaml
```

7. Appliquez le manifeste à votre cluster.

```
kubectl apply -f externalip-webhook.yaml
```

Une tentative de déploiement d'un service sur votre cluster avec une adresse IP spécifiée pour `externalIPs` qui n'est pas contenue dans les blocs que vous avez spécifiés à l'étape Spécifiez des blocs d'adresse CIDR échouera.

Copier une image de conteneur d'un référentiel vers un autre référentiel

Cette rubrique décrit comment extraire une image de conteneur d'un référentiel auquel vos nœuds n'ont pas accès et envoyer l'image vers un référentiel auquel vos nœuds ont accès. Vous pouvez transmettre l'image à Amazon ECR ou à un référentiel alternatif auquel vos nœuds ont accès.

Prérequis

- Le moteur Docker installé et configuré sur votre ordinateur. Pour connaître la marche à suivre, consultez la rubrique Installer Docker Engine dans la documentation Docker.

- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure de l'AWS Command Line Interface (AWS CLI) installée et configurée sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l'AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface. La version d'AWS CLI qui est installée dans AWS CloudShell peut également être en retard de plusieurs versions par rapport à la dernière version. Pour la mettre à jour, consultez Installation d'AWS CLI dans votre répertoire d'accueil dans le Guide de l'utilisateur AWS CloudShell.
- Un point de terminaison d'un VPC d'interface pour Amazon ECR si vous souhaitez que vos nœuds extraient des images de conteneur ou transmettent des images de conteneur vers un référentiel Amazon ECR privé sur le réseau d'Amazon. Pour de plus amples informations, veuillez consulter Créer des points de terminaison de VPC pour Amazon ECR dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Effectuez les étapes suivantes pour extraire une image de conteneur d'un référentiel et la transférer vers votre propre référentiel. Dans les exemples suivants fournis dans cette rubrique, l'image pour l'assistant de métriques Amazon VPC CNI plugin for Kubernetes est extraite. Lorsque vous suivez ces étapes, assurez-vous de remplacer les *`example values`* par vos propres valeurs.

Pour copier une image de conteneur d'un référentiel vers un autre référentiel

1. Si vous ne disposez pas déjà d'un référentiel Amazon ECR ou d'un autre référentiel, créez-en un auquel vos nœuds ont accès. La commande suivante crée un référentiel privé Amazon ECR. Un nom de référentiel privé Amazon ECR doit commencer par une lettre. Il peut uniquement contenir des lettres minuscules, des chiffres, des tirets (-), des traits de soulignement (_) et des barres obliques (/). Pour de plus amples informations, veuillez consulter Créer un référentiel privé dans le Guide de l'utilisateur Amazon Elastic Container Registry.

   Vous pouvez remplacer.*`cni-metrics-helper`* par ce que vous souhaitez. Il est recommandé de créer un référentiel distinct pour chaque image. Nous vous le recommandons, car les identifications d'image doivent être uniques au sein d'un référentiel. Remplacez *`region-code`* par une Région AWS prise en charge par Amazon ECR.

   ```
   aws ecr create-repository --region region-code --repository-name cni-metrics-helper
   ```

2. Déterminez le registre, le référentiel et l'identification (facultatif) de l'image que vos nœuds doivent extraire. Ces informations se trouvent dans le format `registry/repository[:tag]`.

   De nombreuses rubriques Amazon EKS sur l'installation d'images nécessitent que vous appliquiez un fichier manifeste ou que vous installiez l'image à l'aide des Charts de Helm. Cependant, avant d'appliquer un fichier manifeste ou d'installer une charte de Helm, visualisez d'abord le contenu du fichier `values.yaml` manifeste ou de la charte. De cette façon, vous pouvez déterminer le registre, le référentiel et l'identification à extraire.

   Par exemple, vous pouvez trouver la ligne suivante dans le fichier manifeste pour l'assistant de métriques Amazon VPC CNI plugin for Kubernetes. Le registre est `602401143452.dkr.ecr.us-west-2.amazonaws.com`, qui est un registre privé Amazon ECR. Le référentiel est `cni-metrics-helper`.

   ```
   image: "602401143452.dkr.ecr.us-west-2.amazonaws.com/cni-metrics-helper:v1.12.6"
   ```

   Vous pouvez voir les variantes suivantes pour un emplacement d'image :

   - Seul `repository-name:tag`. Dans ce cas, `docker.io` est généralement le registre, mais il n'est pas spécifié puisque Kubernetes l'ajoute par défaut à un nom de référentiel si aucun registre n'est spécifié.
   - `repository-name/repository-namespace/repository:tag`. Un espace de noms de référentiel est facultatif, mais est parfois spécifié par le propriétaire du référentiel pour catégoriser les images. Par exemple, toutes les images Amazon EC2 de la galerie publique Amazon ECR utilisent l'espace de noms `aws-ec2`.

   Avant d'installer une image avec Helm, affichez le fichier `values.yaml` Helm pour déterminer l'emplacement de l'image. Par exemple, le fichier `values.yaml` pour l'assistant de métriques Amazon VPC CNI plugin for Kubernetes comprend les lignes suivantes.

   ```
   image:
     region: us-west-2
     tag: v1.12.6
     account: "602401143452"
     domain: "amazonaws.com"
   ```

3. Extrayez l'image conteneur spécifiée dans le fichier manifeste.

a. Si vous extrayez des images d'un registre public, tel que la galerie publique Amazon ECR, vous pouvez passer à la sous-étape suivante, car l'authentification n'est pas requise. Dans cet exemple, vous vous authentifiez auprès d'un registre privé Amazon ECR contenant le référentiel pour l'image d'assistance des métriques CNI. Amazon EKS conserve l'image dans chaque registre répertorié dans Registres d'images de conteneur Amazon. Vous pouvez vous authentifier auprès de n'importe quel registre en remplaçant *602401143452* et *region-code* par les informations d'un registre différent. Un registre distinct existe pour chaque Région AWS dans laquelle Amazon EKS est pris en charge.

```
aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 602401143452.dkr.ecr.region-code.amazonaws.com
```

b. Extrayez l'image. Dans cet exemple, vous effectuez une extraction à partir du registre auquel vous vous êtes authentifié au cours de la sous-étape précédente. Remplacez *602401143452* et *region-code* par les informations que vous avez spécifiées lors de la sous-étape précédente.

```
docker pull 602401143452.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

4. Étiquetez l'image que vous avez extraite avec votre registre, votre référentiel et votre identification. L'exemple suivant suppose que vous avez extrait l'image du fichier manifeste et que vous allez la transmettre au référentiel privé Amazon ECR que vous avez créé à la première étape. Remplacez *111122223333* par votre ID de compte. Remplacez *region-code* par la Région AWS dans laquelle vous avez créé votre référentiel privé Amazon ECR.

```
docker tag cni-metrics-helper:v1.12.6 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

5. Authentifiez auprès de votre registre. Dans cet exemple, vous vous authentifiez auprès du registre privé Amazon ECR que vous avez créé à la première étape. Pour plus d'informations, veuillez consulter Authentification de registre dans le Guide de l'utilisateur Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 111122223333.dkr.ecr.region-code.amazonaws.com
```

6. Transmettez l'image à votre référentiel. Dans cet exemple, vous transmettez l'image au référentiel privé Amazon ECR que vous avez créé à la première étape. Pour plus d'informations, consultez Transmission d'une image Docker dans le Guide de l'utilisateur Amazon Elastic Container Registry.

   ```
   docker push 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
   ```

7. Mettez à jour le fichier manifeste que vous avez utilisé pour déterminer l'image lors d'une étape précédente avec le `registry/repository:tag` pour l'image que vous avez transmise. Si vous installez avec les Charts de Helm, il existe souvent une option pour spécifier le fichier `registry/repository:tag`. Lors de l'installation du graphique, spécifiez le `registry/repository:tag` pour l'image que vous avez transmise à votre référentiel.

Registres d'images de conteneur Amazon

Lorsque vous déployez des modules complémentaires AWS Amazon EKS dans votre cluster, vos nœuds extraient les images de conteneur requises à partir du registre spécifié dans le mécanisme d'installation du module complémentaire, comme un manifeste d'installation ou un fichier Helm `values.yaml`. Les images sont extraites à partir d'un référentiel privé Amazon EKS Amazon ECR. Amazon EKS réplique les images dans un référentiel dans chaque Région AWS prise en charge par Amazon EKS. Vos nœuds peuvent extraire l'image du conteneur sur Internet à partir de l'un des registres suivants. Vos nœuds peuvent également extraire l'image sur le réseau d'Amazon si vous avez créé un point de terminaison d'un VPC d'interface pour Amazon ECR (AWS PrivateLink) dans votre VPC. Les registres nécessitent une authentification avec un compte IAM AWS. Vos nœuds s'authentifient à l'aide du rôle IAM du nœud Amazon EKS, qui dispose des autorisations dans la politique IAM gérée AmazonEC2ContainerRegistryReadOnly qui lui est associée.

Région AWS	Registre
af-south-1	877085696533.dkr.ecr.af-south-1.amazonaws.com
ap-east-1	800184023465.dkr.ecr.ap-east-1.amazonaws.com

Région AWS	Registre
ap-northeast-1	602401143452.dkr.ecr.ap-northeast-1.amazonaws.com
ap-northeast-2	602401143452.dkr.ecr.ap-northeast-2.amazonaws.com
ap-northeast-3	602401143452.dkr.ecr.ap-northeast-3.amazonaws.com
ap-south-1	602401143452.dkr.ecr.ap-south-1.amazonaws.com
ap-south-2	900889452093.dkr.ecr.ap-south-2.amazonaws.com
ap-southeast-1	602401143452.dkr.ecr.ap-southeast-1.amazonaws.com
ap-southeast-2	602401143452.dkr.ecr.ap-southeast-2.amazonaws.com
ap-southeast-3	296578399912.dkr.ecr.ap-southeast-3.amazonaws.com
ap-southeast-4	491585149902.dkr.ecr.ap-southeast-4.amazonaws.com
ca-central-1	602401143452.dkr.ecr.ca-central-1.amazonaws.com
ca-west-1	761377655185.dkr.ecr.ca-west-1.amazonaws.com
cn-north-1	918309763551.dkr.ecr.cn-north-1.amazonaws.com.cn/
cn-northwest-1	961992271922.dkr.ecr.cn-northwest-1.amazonaws.com.cn/

Région AWS	Registre
eu-central-1	602401143452.dkr.ecr.eu-central-1.amazonaws.com
eu-central-2	900612956339.dkr.ecr.eu-central-2.amazonaws.com
eu-north-1	602401143452.dkr.ecr.eu-north-1.amazonaws.com
eu-south-1	590381155156.dkr.ecr.eu-south-1.amazonaws.com
eu-south-2	455263428931.dkr.ecr.eu-south-2.amazonaws.com
eu-west-1	602401143452.dkr.ecr.eu-west-1.amazonaws.com
eu-west-2	602401143452.dkr.ecr.eu-west-2.amazonaws.com
eu-west-3	602401143452.dkr.ecr.eu-west-3.amazonaws.com
il-central-1	066635153087.dkr.ecr.il-central-1.amazonaws.com
me-south-1	558608220178.dkr.ecr.me-south-1.amazonaws.com
me-central-1	759879836304.dkr.ecr.me-central-1.amazonaws.com
sa-east-1	602401143452.dkr.ecr.sa-east-1.amazonaws.com
us-east-1	602401143452.dkr.ecr.us-east-1.amazonaws.com

Région AWS	Registre
us-east-2	602401143452.dkr.ecr.us-east-2.amazonaws.com
us-gov-east-1	151742754352.dkr.ecr.us-gov-east-1.amazonaws.com
us-gov-west-1	013241004608.dkr.ecr.us-gov-west-1.amazonaws.com
us-west-1	602401143452.dkr.ecr.us-west-1.amazonaws.com
us-west-2	602401143452.dkr.ecr.us-west-2.amazonaws.com

Modules complémentaires Amazon EKS

Un module complémentaire est un logiciel qui fournit des capacités opérationnelles de prise en charge aux applications Kubernetes, mais qui n'est pas spécifique à l'application. Cela inclut des logiciels tels que les agents d'observabilité ou les pilotes Kubernetes qui permettent au cluster d'interagir avec les ressources AWS sous-jacentes pour la mise en réseau, le calcul et le stockage. Les logiciels complémentaires sont généralement conçus et maintenus par la Kubernetes communauté, par des fournisseurs de cloud tels que AWS ou par des fournisseurs tiers. Amazon EKS installe automatiquement des modules complémentaires autogérés tels que le Amazon VPC CNI plugin for Kubernetes, `kube-proxy`, et CoreDNS pour chaque cluster. Vous pouvez modifier la configuration par défaut des modules complémentaires et les mettre à jour quand vous le souhaitez.

Les modules complémentaires Amazon EKS fournissent l'installation et la gestion d'un ensemble organisé de modules complémentaires pour les clusters Amazon EKS. Tous les modules complémentaires Amazon EKS incluent les derniers correctifs de sécurité et corrections de bogues et sont validés par Amazon EKS AWS pour fonctionner avec Amazon EKS. Les modules complémentaires Amazon EKS vous permettent de garantir systématiquement que vos clusters Amazon EKS sont sécurisés et stables. Ils vous permettent aussi de réduire le travail nécessaire pour installer, configurer et mettre à jour les modules complémentaires. Si un module complémentaire autogéré tel que `kube-proxy` est déjà en cours d'exécution sur votre cluster et est disponible en tant

que module complémentaire Amazon EKS, vous pouvez installer le module complémentaire Amazon EKS `kube-proxy` pour commencer à bénéficier des fonctionnalités des modules complémentaires Amazon EKS.

Vous pouvez mettre à jour des champs de configuration spécifiques gérés par Amazon EKS pour les modules complémentaires Amazon EKS via l'API Amazon EKS. Une fois le module complémentaire démarré, vous pouvez également modifier les champs de configuration non gérés par Amazon EKS directement dans le cluster Kubernetes. Cela inclut la définition de champs de configuration spécifiques pour un module complémentaire, le cas échéant. Ces modifications ne sont pas remplacées par Amazon EKS une fois qu'elles sont effectuées. Ceci est rendu possible par la fonction d'application côté serveur de Kubernetes. Pour plus d'informations, consultez Gestion des champs Kubernetes.

Vous pouvez utiliser des modules complémentaires Amazon EKS avec n'importe quel type de nœud Amazon EKS.

Considérations

- Pour configurer des modules complémentaires pour le cluster, votre principal IAM doit disposer des autorisations IAM pour utiliser des modules complémentaires. Pour plus d'informations, consultez les actions ayant `Addon` dans leur nom dans Actions définies par Amazon Elastic Kubernetes Service.
- Les modules complémentaires Amazon EKS s'exécutent sur les nœuds que vous approvisionnez ou configurez pour votre cluster. Les types de nœuds incluent les instances Amazon EC2 et Fargate.
- Vous pouvez modifier les champs qui ne sont pas gérés par Amazon EKS pour personnaliser l'installation d'un module complémentaire Amazon EKS. Pour plus d'informations, consultez Gestion des champs Kubernetes.
- Si vous créez un cluster avec AWS Management Console, les modules complémentaires Amazon EKS `kube-proxy` et CoreDNS Amazon EKS sont automatiquement ajoutés à votre cluster. Amazon VPC CNI plugin for Kubernetes Si vous utilisez `eksctl` pour créer votre cluster avec un fichier `config`, `eksctl` peut également créer le cluster avec les modules complémentaires Amazon EKS. Si vous créez votre cluster à l'aide de `eksctl` sans fichier `config` ou avec un autre outil, les modules complémentaires autogérés `kube-proxy`, Amazon VPC CNI plugin for Kubernetes, et CoreDNS sont installés, plutôt que les modules complémentaires Amazon EKS. Vous pouvez les gérer vous-même ou ajouter manuellement les modules complémentaires Amazon EKS après la création du cluster.

- La ressource `ClusterRoleBinding` `eks:addon-cluster-admin` lie la ressource `ClusterRole` `cluster-admin` à l'identité Kubernetes `eks:addon-manager`. Le rôle dispose des autorisations nécessaires pour que l'identité `eks:addon-manager` crée des espaces de noms Kubernetes et installe des modules complémentaires dans les espaces de noms. Si la ressource `ClusterRoleBinding` `eks:addon-cluster-admin` est supprimée, le cluster Amazon EKS continuera de fonctionner, mais Amazon EKS ne sera plus en mesure de gérer les modules complémentaires. Tous les clusters qui démarrent avec les versions de plateforme suivantes utilisent la nouvelle ressource `ClusterRoleBinding`.

Version de Kubernetes	Version de plateforme EKS
1.20	eks.12
1.21	eks.14
1.22	eks.9
1.23	eks.5
1.24	eks.3

Vous pouvez ajouter, mettre à jour ou supprimer des modules complémentaires Amazon EKS à l'aide de l'API Amazon EKS AWS Management Console AWS CLI,, et`eksctl`. Pour plus d'informations, consultez Gestion des modules complémentaires Amazon EKS. Vous pouvez également créer des modules complémentaires Amazon EKS à l'aide de AWS CloudFormation.

Modules complémentaires Amazon EKS disponibles sur Amazon EKS

Les modules complémentaires Amazon EKS suivants peuvent être créés sur votre cluster. Vous pouvez toujours consulter la liste la plus récente des modules complémentaires disponibles en utilisant`eksctl`, le AWS Management Console, ou le AWS CLI. Pour voir tous les modules complémentaires disponibles ou pour installer un module complémentaire, consultez Création d'un

module complémentaire. Si un module complémentaire nécessite des autorisations IAM, vous devez disposer d'un fournisseur OpenID Connect IAM (OIDC) pour votre cluster. Pour déterminer si vous en avez un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster. Vous pouvez mettre à jour ou supprimer un module complémentaire une fois que vous l'avez installé.

Sélectionnez un module complémentaire pour en savoir plus à son sujet et sur les conditions d'installation de celui-ci.

Amazon VPC CNI plugin for Kubernetes

- Nom – `vpc-cni`
- Description : un plugin d'interface réseau de conteneur (CNI) Kubernetes qui fournit un réseau VPC natif pour votre cluster. Le type autogéré ou géré de ce module complémentaire est installé sur chaque nœud Amazon EC2, par défaut.
- Autorisations IAM requises : ce module complémentaire utilise la fonctionnalité des rôles IAM des comptes de service d'Amazon EKS. Si votre cluster utilise la famille `IPv4`, les autorisations définies dans la politique AmazonEKS_CNI_Policy sont requises. Si votre cluster utilise la famille `IPv6`, vous devez créer une politique IAM avec les autorisations en mode IPv6. Vous pouvez créer un rôle IAM, y attacher l'une des politiques et annoter le compte de service Kubernetes utilisé par le module complémentaire à l'aide de la commande suivante.

  Remplacez *`my-cluster`* par le nom de votre cluster et *`AmazonEKSVPCCNIRole`* par le nom que vous souhaitez pour votre rôle. Si votre cluster utilise la famille `IPv6`, remplacez *`AmazonEKS_CNI_Policy`* par le nom de la politique que vous avez créée. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle, y attacher la politique et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

  ```
  eksctl create iamserviceaccount --name aws-node --namespace kube-system --cluster my-cluster --role-name AmazonEKSVPCCNIRole \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy --approve
  ```

- Informations supplémentaires — Pour en savoir plus sur les paramètres configurables du module complémentaire, consultez aws-vpc-cni-k8s sur. GitHub Pour en savoir plus sur le plugin, voir Proposition : plugin CNI pour la mise en Kubernetes réseau via AWS VPC. Pour plus d'informations sur la création du module complémentaire, consultez la rubrique Création du module complémentaire Amazon EKS.

- Informations sur la mise à jour : vous ne pouvez mettre à jour qu'une seule version mineure à la fois. Par exemple, si votre version actuelle est `1.28.x-eksbuild.y` et que vous voulez la mettre à jour vers `1.30.x-eksbuild.y`, vous devez d'abord mettre à jour votre version actuelle vers `1.29.x-eksbuild.y` et ensuite la mettre à nouveau à jour vers `1.30.x-eksbuild.y`. Pour plus d'informations sur la mise à jour du module complémentaire, consultez la rubrique Mise à jour du module complémentaire Amazon EKS.

CoreDNS

- Nom – `coredns`
- Description : un serveur DNS flexible et extensible qui peut servir de DNS de cluster Kubernetes. Le type autogéré ou géré de ce module complémentaire a été installé par défaut lorsque vous avez créé votre cluster. Lorsque vous lancez un cluster Amazon EKS avec au moins un nœud, deux réplicas de l'image CoreDNS sont déployés par défaut, quel que soit le nombre de nœuds déployés dans votre cluster. Les Pods CoreDNS fournissent une résolution de noms pour tous les Pods du cluster. Vous pouvez déployer les CoreDNS Pods sur les nœuds Fargate si votre cluster comprend un AWS Fargate profil avec un espace de noms qui correspond à l'espace de noms pour les CoreDNS `deployment`.
- Autorisations IAM requises : ce module complémentaire ne nécessite aucune autorisation.
- Informations supplémentaires : pour en savoir plus sur CoreDNS, consultez la section Utilisation de CoreDNS pour la découverte de services (français non garanti) et Personnalisation du service DNS (français non garanti) dans la documentation Kubernetes.

`Kube-proxy`

- Nom – `kube-proxy`
- Description : maintient les règles de réseau sur chaque nœud Amazon EC2. Il permet la communication réseau avec vos Pods. Le type autogéré ou géré de ce module complémentaire est installé par défaut sur chaque nœud Amazon EC2 de votre cluster.
- Autorisations IAM requises : ce module complémentaire ne nécessite aucune autorisation.
- Informations supplémentaires : pour en savoir plus à propos de `kube-proxy`, consultez `kube-proxy` dans la documentation Kubernetes.
- Informations de mise à jour : avant de mettre à jour votre version actuelle, tenez compte des exigences suivantes :

- `Kube-proxy` sur un cluster Amazon EKS possède la même politique de compatibilité et d'inclinaison que Kubernetes.
- `Kube-proxy` doit correspondre à la même version mineure que `kubelet` sur vos nœuds Amazon EC2.
- `Kube-proxy` ne peut pas être postérieur à la version mineure du plan de contrôle de votre cluster.
- La version `kube-proxy` présente sur vos nœuds Amazon EC2 ne peut pas être antérieure de plus de deux versions mineures à votre plan de contrôle. Par exemple, si votre plan de contrôle exécute la version Kubernetes 1.30, la version `kube-proxy` mineure ne peut pas être antérieure à 1.28.
- Si vous avez récemment mis à jour votre cluster vers une nouvelle version mineure de Kubernetes, mettez alors à jour vos nœuds Amazon EC2 vers la même version mineure avant de mettre à jour `kube-proxy` vers la même version mineure que vos nœuds.

Pilote CSI Amazon EBS

- Nom – `aws-ebs-csi-driver`
- Description : un plug-in CSI (Container Storage Interface) Kubernetes qui fournit un stockage Amazon EBS pour votre cluster.
- Autorisations IAM requises : ce module complémentaire utilise la fonctionnalité des rôles IAM des comptes de service d'Amazon EKS. Les autorisations de la politique `AmazonEBSCSIDriverPolicy` AWS gérée sont obligatoires. Vous pouvez créer un rôle IAM et y attacher la politique gérée à l'aide de la commande suivante. Remplacez *`my-cluster`* par le nom de votre cluster et *`AmazonEKS_EBS_CSI_DriverRole`* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil ou une clé KMS personnalisée pour le chiffrement, reportez-vous à la section Création du rôle IAM du pilote CSI Amazon EBS.

```
eksctl create iamserviceaccount \
    --name ebs-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_EBS_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy
 \
```

```
    --approve
```

- Informations supplémentaires : pour en savoir plus à propos de l'add-on, consultez Pilote CSI Amazon EBS.

Pilote CSI Amazon EFS

- Nom – `aws-efs-csi-driver`
- Description : un plugin CSI (Container Storage Interface) Kubernetes qui offre une capacité de stockage Amazon EBS pour votre cluster.
- Autorisations IAM requises : ce module complémentaire utilise la fonctionnalité des rôles IAM des comptes de service d'Amazon EKS. Les autorisations de la politique `AmazonEFSCSIDriverPolicy` AWS gérée sont obligatoires. Vous pouvez créer un rôle IAM et y attacher la politique gérée à l'aide des commandes suivantes. Remplacez *`my-cluster`* par le nom de votre cluster et *`AmazonEKS_EFS_CSI_DriverRole`* par le nom que vous souhaitez pour votre rôle. Ces commandes nécessitent que `eksctl` soit installé sur votre périphérique. Si vous devez utiliser un autre outil, voir Création d'un rôle IAM.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy
 \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

- Informations supplémentaires : pour en savoir plus à propos de l'add-on, consultez Pilote CSI Amazon EFS.

Mountpoint pour le pilote CSI Amazon S3

- Nom – `aws-mountpoint-s3-csi-driver`
- Description : un plugin CSI (Container Storage Interface) Kubernetes qui offre une capacité de stockage Amazon S3 pour votre cluster.
- Autorisations IAM requises : ce module complémentaire utilise la fonctionnalité des rôles IAM des comptes de service d'Amazon EKS. Le rôle IAM créé nécessitera une stratégie d'accès à S3. Suivez les recommandations sur les autorisations IAM de Mountpoint lors de la création de la politique IAM. Vous pouvez également utiliser la politique AWS gérée AmazonS3FullAccess, mais cette politique gérée accorde plus d'autorisations que ce qui est nécessaireMountpoint.

  Vous pouvez créer un rôle IAM et y attacher votre politique à l'aide des commandes suivantes. *Remplacez my-cluster par le nom de votre cluster, region-code par le Région AWS code correct, AmazoneKS_S3_CSI_ par le nom de votre rôle et DriverRoleAmazoneKS_S3_CSI_ _ARN par l'ARN du rôle. DriverRole* Ces commandes nécessitent que `eksctl` soit installé sur votre périphérique. Pour obtenir des instructions sur l'utilisation de la console IAM ou AWS CLI consultezCréation d'un rôle IAM.

  ```
  CLUSTER_NAME=my-cluster
  REGION=region-code
  ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
  POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
  eksctl create iamserviceaccount \
      --name s3-csi-driver-sa \
      --namespace kube-system \
      --cluster $CLUSTER_NAME \
      --attach-policy-arn $POLICY_ARN \
      --approve \
      --role-name $ROLE_NAME \
      --region $REGION \
      --role-only
  ```

- Informations supplémentaires : pour en savoir plus à propos de l'add-on, consultez Mountpoint pour le pilote CSI Amazon S3.

Contrôleur d'instantané CSI

- Nom – `snapshot-controller`

- Description : le contrôleur d'instantané de l'interface de stockage de conteneurs (CSI) permet d'utiliser la fonctionnalité d'instantané dans les pilotes CSI compatibles, tels que le pilote CSI Amazon EBS.
- Autorisations IAM requises : ce module complémentaire ne nécessite aucune autorisation.
- Informations supplémentaires : pour en savoir plus à propos de l'add-on, consultez Contrôleur d'instantané CSI.

AWS Distro pour OpenTelemetry

- Nom – `adot`
- Description — The AWS Distro for OpenTelemetry (ADOT) est une distribution sécurisée, prête pour la production et AWS prise en charge du projet. OpenTelemetry
- Autorisations IAM requises : ce module complémentaire ne nécessite des autorisations IAM que si vous utilisez l'une des ressources personnalisées préconfigurées pour lesquelles il est possible d'accéder via la configuration avancée.
- Informations supplémentaires — Pour plus d'informations, consultez Getting Started with AWS Distro pour OpenTelemetry l'utilisation des modules complémentaires EKS dans la AWS distribution pour OpenTelemetry obtenir de la documentation.

  ADOT exige que `cert-manager` soit déployé sur le cluster en tant que condition préalable, sinon ce module complémentaire ne fonctionnera pas s'il est déployé directement à l'aide de la propriété `cluster_addons` d'Amazon EKS Terraform. Pour plus d'informations sur les exigences, reportez-vous à la section Conditions requises pour démarrer avec AWS Distro pour OpenTelemetry l'utilisation des modules complémentaires EKS dans la AWS distribution pour OpenTelemetry obtenir de la documentation.

Amazon GuardDuty agent

- Nom – `aws-guardduty-agent`
- Description — Amazon GuardDuty est un service de surveillance de la sécurité qui analyse et traite les sources de données fondamentales, notamment les événements AWS CloudTrail de gestion et les journaux de flux Amazon VPC. Amazon traite GuardDuty également des fonctionnalités, telles que les journaux Kubernetes d'audit et la surveillance du temps d'exécution.
- Autorisations IAM requises : ce module complémentaire ne nécessite aucune autorisation.

- Informations supplémentaires — Pour plus d'informations, consultez la section Surveillance du temps d'exécution pour les clusters Amazon EKS sur Amazon GuardDuty.
  - Pour détecter les menaces de sécurité potentielles dans vos clusters Amazon EKS, activez la surveillance de l' GuardDuty exécution Amazon et déployez l'agent de GuardDuty sécurité sur vos clusters Amazon EKS.

Agent Amazon CloudWatch Observability

- Nom – `amazon-cloudwatch-observability`
- Description Amazon CloudWatch Agent est le service de surveillance et d'observabilité fourni par AWS. Ce module complémentaire installe l' CloudWatch agent et active à la fois CloudWatch Application Signals et CloudWatch Container Insights avec une observabilité améliorée pour Amazon EKS.
- Autorisations IAM requises : ce module complémentaire utilise la fonctionnalité des rôles IAM des comptes de service d'Amazon EKS. Les autorisations indiquées dans les CloudWatchAgentServerpolitiques AWS gérées par AWSXrayWriteOnlyAccesset Policy sont obligatoires. Vous pouvez créer un rôle IAM, y attacher les politiques gérées et annoter le compte de service Kubernetes utilisé par le module complémentaire à l'aide de la commande suivante. Remplacez *`my-cluster`* par le nom de votre cluster et *`AmazonEKS_Observability_role`* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle, y attacher la politique et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

```
eksctl create iamserviceaccount \
    --name cloudwatch-agent \
    --namespace amazon-cloudwatch \
    --cluster my-cluster \
    --role-name AmazonEKS_Observability_Role \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess \
    --attach-policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy \
    --approve
```

- Informations supplémentaires — Pour plus d'informations, voir Installer l' CloudWatch agent.

Agent d'identité du pod Amazon EKS

- Nom – `eks-pod-identity-agent`
- Description — Amazon EKS Pod Identity permet de gérer les informations d'identification de vos applications, de la même manière que les profils d' Amazon EC2 instance fournissent des informations d'identification aux instances EC2.
- Autorisations IAM requises : ce module complémentaire utilise les autorisations de Rôle IAM de nœud Amazon EKS.
- Informations sur la mise à jour : vous ne pouvez mettre à jour qu'une seule version mineure à la fois. Par exemple, si votre version actuelle est `1.28.`*x*`-eksbuild.`*y* et que vous voulez la mettre à jour vers `1.30.`*x*`-eksbuild.`*y*, vous devez d'abord mettre à jour votre version actuelle vers `1.29.`*x*`-eksbuild.`*y* et ensuite la mettre à nouveau à jour vers `1.30.`*x*`-eksbuild.`*y*. Pour plus d'informations sur la mise à jour du module complémentaire, consultez la rubrique Mise à jour du module complémentaire Amazon EKS.

Modules complémentaires Amazon EKS proposés par des fournisseurs de logiciels indépendants

Outre la liste précédente de modules complémentaires Amazon EKS, vous pouvez également ajouter une large sélection de modules complémentaires Amazon EKS pour logiciels opérationnels proposés par des fournisseurs indépendants de logiciels. Sélectionnez un module complémentaire pour en savoir plus à son sujet et sur les conditions d'installation de celui-ci.

Trouvez, procurez-vous et déployez des modules complémentaires depuis AWS Marketplace vers Amazon EKS (YouTube).

Accuknox

- Éditeur – Accuknox
- Nom – `accuknox_kubearmor`
- Espace de noms – `kubearmor`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.

- Instructions de configuration et d'utilisation : voir Getting Started with KubeArmor dans la KubeArmor documentation.

Akuity

- Éditeur – Akuity
- Nom – `akuity_agent`
- Espace de noms – `akuity`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez la section Installation de l'agent Akuity sur Amazon EKS avec le module complémentaire Akuity EKS dans la documentation de la plateforme Akuity.

Calyptia

- Éditeur – Calyptia
- Nom – `calyptia_fluent-bit`
- Espace de noms – `calytia-fluentbit`
- Nom du compte de service – `clyptia-fluentbit`
- AWS politique IAM gérée — AWSMarketplaceMeteringRegisterUsage.
- Commande permettant de créer le rôle IAM requis – Pour utiliser la commande suivante, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster et *`my-calyptia-role`* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

```
eksctl create iamserviceaccount --name service-account-name  --namespace calyptia-
fluentbit --cluster my-cluster --role-name my-calyptia-role \
```

```
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/
AWSMarketplaceMeteringRegisterUsage --approve
```

- Instructions de configuration et d'utilisation : consultez Calyptia for Fluent Bit dans la documentation de Calyptia.

Cisco Observability Collector

- Éditeur – Cisco
- Nom – `cisco_cisco-cloud-observability-collectors`
- Espace de noms – `appdynamics`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Consultez la section Utiliser les modules complémentaires AWS Marketplace de Cisco Cloud Observability dans la AppDynamics documentation Cisco.

Cisco Observability Operator

- Éditeur – Cisco
- Nom – `cisco_cisco-cloud-observability-operators`
- Espace de noms – `appdynamics`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Consultez la section Utiliser les modules complémentaires AWS Marketplace de Cisco Cloud Observability dans la AppDynamics documentation Cisco.

CLOUDSOFT

- Éditeur – CLOUDSOFT
- Nom – `cloudsoft_cloudsoft-amp`
- Espace de noms – `cloudsoft-amp`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Consultez Amazon EKS ADDON dans la documentation CLOUDSOFT.

Cribl

- Éditeur – Cribl
- Nom – `cribl_cribledge`
- Espace de noms – `cribledge`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez Installation du module complémentaire Cribl Amazon EKS pour Edge dans la documentation Cribl.

Dynatrace

- Éditeur – Dynatrace
- Nom – `dynatrace_dynatrace-operator`
- Espace de noms – `dynatrace`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.

- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation – Consultez Surveillance de Kubernetes dans la documentation dynatrace.

Datree

- Éditeur – Datree
- Nom – `datree_engine-pro`
- Espace de noms – `datree`
- Nom du compte de service – datree-webhook-server-awsmp
- AWS politique IAM gérée — AWSLicenseManagerConsumptionPolicy.
- Commande permettant de créer le rôle IAM requis – Pour utiliser la commande suivante, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster et *`my-datree-role`* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

  ```
  eksctl create iamserviceaccount --name datree-webhook-server-awsmp --namespace datree
   --cluster my-cluster --role-name my-datree-role \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
  AWSLicenseManagerConsumptionPolicy --approve
  ```

- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : Consultez Intégration d'Amazon EKS dans la documentation Datree.

Datadog

- Éditeur – Datadog
- Nom – `datadog_operator`

- Espace de noms – `datadog-agent`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez Installation de l'agent Datadog sur Amazon EKS avec le module complémentaire Datadog Operator dans la documentation Datadog.

Groundcover

- Éditeur – groundcover
- Nom – `groundcover_agent`
- Espace de noms – `groundcover`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez Installation du module complémentaire Amazon EKS groundcover dans la documentation groundcover.

Grafana Labs

- Éditeur – Grafana Labs
- Nom – `grafana-labs_kubernetes-monitoring`
- Espace de noms – `monitoring`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.

- Instructions de configuration et d'utilisation : consultez la section Configure Kubernetes Monitoring as an Add-on with Amazon EKS (Configurer Kubernetes Monitoring en tant que module complémentaire avec Amazon EKS) dans la documentation Grafana Labs.

HA Proxy

- Éditeur – HA Proxy
- Nom – `haproxy-technologies_kubernetes-ingress-ee`
- Espace de noms – `haproxy-controller`
- Nom du compte de service – `customer defined`
- AWS politique IAM gérée — AWSLicenseManagerConsumptionPolicy.
- Commande permettant de créer le rôle IAM requis – Pour utiliser la commande suivante, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster et *`my-haproxy-role`* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

```
eksctl create iamserviceaccount --name service-account-name  --namespace haproxy-
controller --cluster my-cluster --role-name my-haproxy-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation – Voir la rubrique Installation du contrôleur d'entrée Kubernetes HAProxy Enterprise sur Amazon EKS à partir de AWS dans la documentation HAProxy.

Kpow

- Éditeur – Factorhouse
- Nom – `factorhouse_kpow`
- Espace de noms – `factorhouse`

- Nom du compte de service – `kpow`
- AWS politique IAM gérée — AWSLicenseManagerConsumptionPolicy
- Commande permettant de créer le rôle IAM requis – Pour utiliser la commande suivante, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster. Remplacez *my-cluster* par le nom de votre cluster et *my-kpow-role* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

  ```
  eksctl create iamserviceaccount --name kpow --namespace factorhouse --cluster my-cluster --role-name my-kpow-role \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/AWSLicenseManagerConsumptionPolicy --approve
  ```

- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation – Consultez AWS Marketplace LM dans la documentation Kpow.

Kubecost

- Éditeur – Kubecost
- Nom – `kubecost_kubecost`
- Espace de noms – `kubecost`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez la section Intégration de la facturation dans le AWS cloud dans la Kubecost documentation.
- Si vous disposez d'un cluster correspondant à la version `1.23` ou ultérieure, the section called "Pilote CSI Amazon EBS" doit y être installé. Sinon, vous recevrez une erreur.

Kasten

- Éditeur – Kasten by Veeam
- Nom – `kasten_k10`
- Espace de noms – `kasten-io`
- Nom du compte de service – `k10-k10`
- AWS politique IAM gérée — AWSLicenseManagerConsumptionPolicy.
- Commande permettant de créer le rôle IAM requis – Pour utiliser la commande suivante, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un, ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster et *`my-kasten-role`* par le nom que vous souhaitez pour votre rôle. Pour pouvoir utiliser cette commande, `eksctl` doit être installé sur votre appareil. Si vous devez utiliser un autre outil pour créer le rôle et annoter le compte de service Kubernetes, consultez Configurer un compte Kubernetes de service pour qu'il assume un rôle IAM.

  ```
  eksctl create iamserviceaccount --name k10-k10 --namespace kasten-io --cluster my-cluster --role-name my-kasten-role \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/AWSLicenseManagerConsumptionPolicy --approve
  ```

- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Consultez la section Installation du K10 à l' AWS aide du module complémentaire Amazon EKS dans la documentation de Kasten.
- Informations supplémentaires : Si votre cluster Amazon EKS est une version Kubernetes `1.23` ou une version ultérieure, le pilote CSI Amazon EBS doit être installé sur votre cluster avec une version `StorageClass` par défaut.

Kong

- Éditeur – Kong
- Nom – `kong_konnect-ri`
- Espace de noms – `kong`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.

- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions d'installation et d'utilisation : consultez Installation du module complémentaire Kong Gateway EKS dans la documentation Kong.

LeakSignal

- Éditeur – LeakSignal
- Nom – `leaksignal_leakagent`
- Espace de noms – `leakagent`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Voir Installer le LeakAgent module complémentaire dans la LeakSignal documentation.

NetApp

- Éditeur – NetApp
- Nom – `netapp_trident-operator`
- Espace de noms – `trident`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Voir Configuration du module complémentaire Astra Trident EKS dans la NetApp documentation.

New Relic

- Éditeur – New Relic
- Nom – `new-relic_kubernetes-operator`
- Espace de noms – `newrelic`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez Installation du module complémentaire New Relic pour EKS dans la documentation New Relic.

Rafay

- Éditeur – Rafay
- Nom – `rafay-systems_rafay-operator`
- Espace de noms – `rafay-system`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez Installation du module complémentaire Amazon EKS Rafay dans la documentation Rafay.

Solo.io

- Éditeur – Solo.io
- Nom – `solo-io_istio-distro`
- Espace de noms – `istio-system`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.

- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Consultez la section Installation d'Istio dans la documentation de Solo.io.

Stormforge

- Éditeur – Stormforge
- Nom – `stormforge_optimize-Live`
- Espace de noms – `stormforge-system`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : voir Installation de l' StormForge agent dans la StormForge documentation.

Splunk

- Éditeur – Splunk
- Nom – `splunk_splunk-otel-collector-chart`
- Espace de noms – `splunk-monitoring`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation : consultez Install the Splunk add-on for Amazon EKS (Installation du module complémentaire Splunk pour Amazon EKS) dans la documentation Splunk.

Teleport

- Éditeur – Teleport

- Nom – `teleport_teleport`
- Espace de noms – `teleport`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation – Consultez Fonctionnement de Teleport dans la documentation Teleport.

Tetrate

- Editeur – Tetrate Io
- Nom – `tetrate-io_istio-distro`
- Espace de noms – `istio-system`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation – Consultez le site Internet de Tetrate Istio Distro.

Upbound Universal Crossplane

- Éditeur – Upbound
- Nom – `upbound_universal-crossplane`
- Espace de noms – `upbound-system`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.

- Instructions de configuration et d'utilisation – Consultez Upbound Universal Crossplane (UXP) dans la documentation Upbound.

Upwind

- Éditeur – Upwind
- Nom – `upwind`
- Espace de noms – `upwind`
- Nom du compte de service – Aucun compte de service n'est utilisé avec ce module complémentaire.
- AWS stratégie IAM gérée : aucune stratégie gérée n'est utilisée avec ce module complémentaire.
- Autorisations IAM personnalisées – Aucune autorisation personnalisée n'est utilisée avec ce module complémentaire.
- Instructions de configuration et d'utilisation — Consultez les étapes d'installation dans la documentation Upwind.

Gestion des modules complémentaires Amazon EKS

Les modules complémentaires Amazon EKS sont un ensemble organisé de modules complémentaires destinés aux clusters Amazon EKS. Tous les modules complémentaires Amazon EKS :

- incluent les derniers correctifs de sécurité et corrections de bogues.
- sont validés par AWS pour fonctionner avec Amazon EKS.
- réduisent la quantité de travail requise pour gérer le module complémentaire.

Vous AWS Management Console avertit lorsqu'une nouvelle version est disponible pour un module complémentaire Amazon EKS. Vous pouvez simplement lancer la mise à jour, après quoi Amazon EKS met à jour le module complémentaire pour vous.

Pour obtenir la liste des modules complémentaires disponibles, consultez Modules complémentaires Amazon EKS disponibles sur Amazon EKS. Pour plus d'informations sur la gestion des champs Kubernetes, consultez Gestion des champs Kubernetes.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS.

Création d'un module complémentaire

Vous pouvez créer un module complémentaire Amazon EKS à l'aide du `eksctl` AWS Management Console, ou du AWS CLI. Si le module complémentaire exige un rôle IAM, consultez les détails du module complémentaire spécifique dans Modules complémentaires Amazon EKS disponibles sur Amazon EKS pour obtenir des détails sur la création du rôle.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour créer un module complémentaire Amazon EKS en utilisant **`eksctl`**

1. Affichez les noms des modules complémentaires disponibles pour une version de cluster. Remplacez *`1.30`* par la version de votre cluster.

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 | grep AddonName
```

L'exemple qui suit illustre un résultat.

```
"AddonName": "aws-ebs-csi-driver",
                        "AddonName": "coredns",
                        "AddonName": "kube-proxy",
                        "AddonName": "vpc-cni",
                        "AddonName": "adot",
                        "AddonName": "dynatrace_dynatrace-operator",
                        "AddonName": "upbound_universal-crossplane",
                        "AddonName": "teleport_teleport",
                        "AddonName": "factorhouse_kpow",
                        [...]
```

2. Consultez les versions disponibles pour le module complémentaire que vous souhaitez créer. Remplacez *`1.30`* par la version de votre cluster. Remplacez *`name-of-addon`* par le nom du

module complémentaire dont vous souhaitez afficher les versions. Ce nom doit correspondre à l'un des noms renvoyés lors des étapes précédentes.

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 --name name-of-addon | grep AddonVersion
```

La sortie suivante est un exemple de ce qui est renvoyé pour le module complémentaire nommé `vpc-cni`. Vous pouvez voir que le module complémentaire dispose de plusieurs versions disponibles.

```
"AddonVersions": [
    "AddonVersion": "v1.12.0-eksbuild.1",
    "AddonVersion": "v1.11.4-eksbuild.1",
    "AddonVersion": "v1.10.4-eksbuild.1",
    "AddonVersion": "v1.9.3-eksbuild.1",
```

3. Déterminez si le module complémentaire que vous souhaitez créer est un module Amazon EKS ou un module complémentaire AWS Marketplace . AWS Marketplace Il contient des modules complémentaires tiers qui vous obligent à effectuer des étapes supplémentaires pour créer le module complémentaire.

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 --name name-of-addon | grep ProductUrl
```

Si aucune sortie n'est renvoyée, le module complémentaire est un module Amazon EKS. Si une sortie est renvoyée, le module complémentaire est un AWS Marketplace module complémentaire. La sortie suivante concerne un module complémentaire nommé `teleport_teleport`.

```
"ProductUrl": "https://aws.amazon.com/marketplace/pp?
sku=3bda70bb-566f-4976-806c-f96faef18b26"
```

Pour en savoir plus sur le module complémentaire, AWS Marketplace consultez l'URL renvoyée. Si le module complémentaire nécessite un abonnement, vous pouvez vous y abonner via le AWS Marketplace. Si vous souhaitez créer un module complémentaire à partir du AWS Marketplace, le principal IAM que vous utilisez pour créer le module complémentaire doit être autorisé à créer le rôle lié au AWSServiceRoleForAWSLicenseManagerRoleservice. Pour plus d'informations sur l'attribution d'autorisations à une entité IAM, consultez la rubrique

Ajout et suppression d'autorisations basées sur l'identité IAM dans le Guide de l'utilisateur IAM.

4. Créez un module complémentaire Amazon EKS. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée :

   - Remplacez *my-cluster* par le nom de votre cluster.
   - Remplacez *name-of-addon* par le nom du module complémentaire que vous souhaitez créer.
   - Si vous souhaitez une version du module complémentaire antérieure à la dernière version, remplacez *latest* par le numéro de version renvoyé dans la sortie d'une étape précédente que vous souhaitez utiliser.
   - Si le module complémentaire utilise un rôle de compte de service, remplacez *111122223333* par votre ID de compte et *role-name* par le nom du rôle. Pour obtenir des instructions sur la création d'un rôle pour votre compte de service, consultez la documentation du module complémentaire que vous créez. Pour spécifier un rôle de compte de service, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.

     Si le module complémentaire n'utilise aucun rôle de compte de service, supprimez ***--service-account-role-arn*** **arn:aws:iam::*111122223333*:role/*role-name***.
   - Cet exemple de commande remplace la configuration de toute version autogérée existante du module complémentaire, le cas échéant. Si vous ne souhaitez pas remplacer la configuration d'un module complémentaire autogéré existant, supprimez l'option *--force*. Si vous supprimez cette option et que le module complémentaire doit remplacer la configuration d'un module complémentaire autogéré existant, la création du module complémentaire Amazon EKS échoue et vous recevez un message d'erreur pour vous aider à résoudre le conflit. Avant de spécifier cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer, car ces paramètres sont remplacés par cette option.

     ```
     eksctl create addon --cluster my-cluster --name name-of-addon --version latest
      \
         --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
     force
     ```

Vous pouvez consulter la liste complète des options disponibles pour la commande.

```
eksctl create addon --help
```

Pour plus d'informations sur les options disponibles, consultez Modules complémentaires dans la documentation `eksctl`.

AWS Management Console

Pour créer un module complémentaire Amazon EKS à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez créer le module complémentaire.
3. Choisissez l'onglet Modules complémentaires.
4. Choisissez Obtenez plus de modules complémentaires.
5. Choisissez les modules complémentaires que vous souhaitez ajouter à votre cluster. Vous pouvez ajouter autant de modules complémentaires Amazon EKS et de modules complémentaires AWS Marketplace que vous le souhaitez.

   Pour les AWS Marketplacemodules complémentaires, le principal IAM que vous utilisez pour créer le module complémentaire doit être autorisé à lire les droits relatifs au module complémentaire depuis le. AWS LicenseManager AWS LicenseManager nécessite un rôle AWSServiceRoleForAWSLicenseManagerRolelié au service (SLR) qui permet aux AWS ressources de gérer les licences en votre nom. Le rôle SLR est une exigence unique, par compte, et vous n'avez pas à créer de rôles SLR distincts pour chaque module complémentaire ni pour chaque cluster. Pour plus d'informations sur l'attribution d'autorisations à un principal IAM, consultez la rubrique Ajout et suppression d'autorisations basées sur l'identité IAM du Guide de l'utilisateur IAM.

   Si les modules complémentaires AWS Marketplace que vous souhaitez installer ne figurent pas dans la liste, vous pouvez rechercher les modules complémentaires disponibles en saisissant du texte dans la zone de recherche. Dans les Options de filtrage, vous pouvez également filtrer par catégorie, fournisseur, ou modèle de tarification puis choisir les modules complémentaires dans les résultats de la recherche. Après avoir sélectionné les modules complémentaires que vous souhaitez installer, sélectionnez Next (Suivant).

6. Sur la page Configure selected add-ons settings (Configurer les paramètres des modules complémentaires sélectionnés) :

   - Choisissez Afficher les options d'abonnement pour ouvrir le formulaire des Options d'abonnement. Passez en revue les sections Détails de la tarification et Mentions légales, puis cliquez sur le bouton S'abonner pour continuer.
   - Pour Version, sélectionnez la version à installer. Nous recommandons la version marquée comme la plus récente, à moins que le module complémentaire que vous créez ne recommande une version différente. Pour déterminer si la version d'un module complémentaire est recommandée, consultez la documentation du module complémentaire que vous créez.
   - Si tous les modules complémentaires que vous avez sélectionnés indiquent Requires subscription (Nécessite un abonnement) sous Status (État), sélectionnez Next (Suivant). Vous ne pouvez pas configurer davantage ces modules complémentaires tant que vous ne vous y êtes pas abonné après la création de votre cluster. Pour les modules complémentaires qui n'indiquent pas Requires subscription (Nécessite un abonnement) sous Status (État) :
     - Pour Select IAM role (Sélectionner un rôle IAM), acceptez l'option par défaut, sauf si le module complémentaire nécessite des autorisations IAM. Si le module complémentaire nécessite AWS des autorisations, vous pouvez utiliser le rôle IAM du nœud (Non défini) ou un rôle existant que vous avez créé pour être utilisé avec le module complémentaire. S'il n'y a aucun rôle à sélectionner, vous n'avez pas de rôle existant. Quelle que soit l'option choisie, consultez la documentation du module complémentaire que vous créez pour créer une politique IAM et l'associer à un rôle. Pour sélectionner un rôle IAM, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
     - Sélectionnez Optional configuration settings (Paramètres de configuration facultatifs).
       - Si le module complémentaire nécessite une configuration, saisissez-la dans la zone Configuration values (Valeurs de configuration). Pour déterminer si le module complémentaire nécessite des informations de configuration, consultez la documentation du module complémentaire que vous créez.
       - Sélectionnez l'une des options disponibles pour Conflict resolution method (Méthode de résolution des conflits).
     - Choisissez Suivant.

7. Sur la page Vérifier et ajouter, choisissez Créer. Une fois l'installation des modules complémentaire terminée, vous pouvez voir les modules complémentaires installés.
8. Si l'un des modules complémentaires que vous avez installés nécessite un abonnement, procédez comme suit :

    1. Cliquez sur le bouton Subscribe (S'abonner) dans le coin inférieur droit du module complémentaire. Vous êtes redirigé vers la page du module complémentaire dans le AWS Marketplace. Lisez les informations relatives au module complémentaire, telles que Product Overview (Présentation du produit) et Pricing Information (Informations sur la tarification).
    2. Cliquez sur le bouton Continue to Subscribe (Continuer pour s'abonner) en haut à droite de la page du module complémentaire.
    3. Lisez les Conditions générales. Si vous les acceptez, sélectionnez Accept Terms (Accepter les conditions). Le traitement de l'abonnement peut prendre plusieurs minutes. Pendant le traitement de l'abonnement, le bouton Return to Amazon EKS Console (Retourner à la console Amazon EKS) est grisé.
    4. Une fois le traitement de l'abonnement terminé, le bouton Return to Amazon EKS Console (Retourner à la console Amazon EKS) n'est plus grisé. Cliquez sur le bouton pour revenir à l'onglet Add-ons (Modules complémentaires) de la console Amazon EKS pour votre cluster.
    5. Pour le module complémentaire auquel vous êtes abonné, sélectionnez Remove and reinstall (Supprimer et réinstaller), puis Reinstall add-on (Réinstaller le module complémentaire). L'installation du module complémentaire peut prendre plusieurs minutes. Une fois l'installation terminée, vous pouvez configurer le module complémentaire.

AWS CLI

Prérequis

Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version

installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

Pour créer un module complémentaire Amazon EKS à l'aide du AWS CLI

1. Déterminez quels modules complémentaires sont disponibles. Vous pouvez voir tous les modules complémentaires disponibles, leur type et leur éditeur. Vous pouvez également consulter l'URL des modules complémentaires disponibles via le AWS Marketplace. Remplacez *1.30* par la version de votre cluster.

```
aws eks describe-addon-versions --kubernetes-version 1.30 \
    --query 'addons[].{MarketplaceProductUrl: marketplaceInformation.productUrl,
 Name: addonName, Owner: owner Publisher: publisher, Type: type}' --output table
```

L'exemple qui suit illustre un résultat.

```
-------------------------------------------------------------------------------------------
|
 DescribeAddonVersions
       |
+-----------------------------------------------------------------
+-------------------------------+------------------+--------------
+---------------------+
|                      MarketplaceProductUrl                      |
 Name              |      Owner       |  Publisher   |        Type         |
+-----------------------------------------------------------------
+-------------------------------+------------------+--------------
+---------------------+
|  None                                                           |  aws-ebs-csi-
driver           |  aws             |  eks         |  storage            |
|  None                                                           |  coredns
                |  aws             |  eks         |  networking         |
|  None                                                           |  kube-proxy
                |  aws             |  eks         |  networking         |
|  None                                                           |  vpc-cni
                |  aws             |  eks         |  networking         |
|  None                                                           |  adot
                |  aws             |  eks         |  observability      |
|  https://aws.amazon.com/marketplace/pp/prodview-brb73nceicv7u |
 dynatrace_dynatrace-operator |  aws-marketplace |  dynatrace   |  monitoring
      |
```

```
|  https://aws.amazon.com/marketplace/pp/prodview-uhc2iwi5xysoc |
 upbound_universal-crossplane |  aws-marketplace |  upbound     |  infra-
management   |
|  https://aws.amazon.com/marketplace/pp/prodview-hd2ydsrgqy4li |
 teleport_teleport            |  aws-marketplace |  teleport    |  policy-
management  |
|  https://aws.amazon.com/marketplace/pp/prodview-vgghgqdsplhvc |
 factorhouse_kpow             |  aws-marketplace |  factorhouse |  monitoring
      |
|  [...]                                                       |  [...]
                |  [...]           |  [...]       |  [...]              |
+-------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
```

Votre sortie peut être différente. Dans cet exemple de sortie, trois modules complémentaires différents sont disponibles par type `networking` et cinq modules complémentaires avec un éditeur de type `eks`. Les modules complémentaires avec `aws-marketplace` dans la colonne `0wner` peuvent nécessiter un abonnement avant d'être installés. Vous pouvez consulter l'URL pour en savoir plus sur le module complémentaire et vous y abonner.

2. Vous pouvez voir quelles versions sont disponibles pour chaque module complémentaire. Remplacez *`1.30`* par la version de votre cluster et remplacez *`vpc-cni`* par le nom d'un module complémentaire renvoyé à l'étape précédente.

```
aws eks describe-addon-versions --kubernetes-version 1.30 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

L'exemple qui suit illustre un résultat.

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
|  True           |  v1.11.4-eksbuild.1  |
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

La version avec `True` dans la colonne `Defaultversion` correspond à la version avec laquelle le module complémentaire est créé, par défaut.

3. (Facultatif) Recherchez les options de configuration du module complémentaire de votre choix en exécutant la commande suivante :

```
aws eks describe-addon-configuration --addon-name vpc-cni --addon-
version v1.12.0-eksbuild.1
```


```
{
    "addonName": "vpc-cni",
    "addonVersion": "v1.12.0-eksbuild.1",
    "configurationSchema": "{\"$ref\":\"#/definitions/VpcCni\",\"$schema
\":\"http://json-schema.org/draft-06/schema#\",\"definitions\":{\"Cri\":
{\"additionalProperties\":false,\"properties\":{\"hostPath\":{\"$ref\":
\"#/definitions/HostPath\"}},\"title\":\"Cri\",\"type\":\"object\"},\"Env
\":{\"additionalProperties\":false,\"properties\":{\"ADDITIONAL_ENI_TAGS
\":{\"type\":\"string\"},\"AWS_VPC_CNI_NODE_PORT_SUPPORT\":{\"format\":
\"boolean\",\"type\":\"string\"},\"AWS_VPC_ENI_MTU\":{\"format\":\"integer
\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CONFIGURE_RPFILTER\":{\"format
\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG\":
{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_EXTERNALSNAT
\":{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOGLEVEL
\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOG_FILE\":{\"type
\":\"string\"},\"AWS_VPC_K8S_CNI_RANDOMIZESNAT\":{\"type\":
\"string\"},\"AWS_VPC_K8S_CNI_VETHPREFIX\":{\"type\":\"string
\"},\"AWS_VPC_K8S_PLUGIN_LOG_FILE\":{\"type\":\"string\"},
\"AWS_VPC_K8S_PLUGIN_LOG_LEVEL\":{\"type\":\"string\"},\"DISABLE_INTROSPECTION
\":{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_METRICS\":{\"format
\":\"boolean\",\"type\":\"string\"},\"DISABLE_NETWORK_RESOURCE_PROVISIONING
\":{\"format\":\"boolean\",\"type\":\"string\"},\"ENABLE_POD_ENI\":{\"format
\":\"boolean\",\"type\":\"string\"},\"ENABLE_PREFIX_DELEGATION\":{\"format
\":\"boolean\",\"type\":\"string\"},\"WARM_ENI_TARGET\":{\"format\":\"integer
\",\"type\":\"string\"},\"WARM_PREFIX_TARGET\":{\"format\":\"integer\",
\"type\":\"string\"}},\"title\":\"Env\",\"type\":\"object\"},\"HostPath\":
{\"additionalProperties\":false,\"properties\":{\"path\":{\"type\":\"string\"}},
\"title\":\"HostPath\",\"type\":\"object\"},\"Limits\":{\"additionalProperties
\":false,\"properties\":{\"cpu\":{\"type\":\"string\"},\"memory\":{\"type
\":\"string\"}},\"title\":\"Limits\",\"type\":\"object\"},\"Resources\":
{\"additionalProperties\":false,\"properties\":{\"limits\":{\"$ref\":\"#/
definitions/Limits\"},\"requests\":{\"$ref\":\"#/definitions/Limits\"}},
\"title\":\"Resources\",\"type\":\"object\"},\"VpcCni\":{\"additionalProperties
```

```
\":false,\"properties\":{\"cri\":{\"$ref\":\"#/definitions/Cri\"},\"env\":
{\"$ref\":\"#/definitions/Env\"},\"resources\":{\"$ref\":\"#/definitions/
Resources\"}},\"title\":\"VpcCni\",\"type\":\"object\"}}}"
}
```

La sortie correspond à un schéma JSON standard.

Voici un exemple de valeurs de configuration valides, au format JSON, fonctionnant avec le schéma ci-dessus.

```
{
  "resources": {
    "limits": {
      "cpu": "100m"
    }
  }
}
```

Voici un exemple de valeurs de configuration valides, au format YAML, fonctionnant avec le schéma ci-dessus.

```
  resources:
    limits:
      cpu: 100m
```

4. Déterminez si le module complémentaire nécessite des autorisations IAM. Dans ce cas, vous devez (1) déterminer si vous souhaitez utiliser EKS Pod Identities ou IAM Roles for Service Accounts (IRSA), (2) déterminer l'ARN du rôle IAM à utiliser avec le module complémentaire et (3) déterminer le nom du compte de service Kubernetes utilisé par le module complémentaire. Vous pouvez trouver ces informations dans la documentation ou à l'aide de l' AWS API, voir Récupérer les informations IAM relatives à un module complémentaire.

   - Amazon EKS suggère d'utiliser EKS Pod Identities si le module complémentaire le prend en charge. Cela nécessite que l'agent Pod Identity soit installé sur votre cluster. Pour plus d'informations sur l'utilisation de Pod Identities avec des modules complémentaires, consultezAssocier un rôle IAM à un module complémentaire Amazon EKS à l'aide de Pod Identity.

- Si le module complémentaire ou votre cluster n'est pas configuré pour EKS Pod Identities, utilisez IRSA. Vérifiez que l'IRSA est configuré sur votre cluster.
- Consultez la documentation relative aux modules complémentaires Amazon EKS pour déterminer si le module complémentaire nécessite des autorisations IAM et le nom du compte de service Kubernetes associé.

5. Créez un module complémentaire Amazon EKS. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée :

   - Remplacez *my-cluster* par le nom de votre cluster.
   - Remplacez *vpc-cni* par un nom de module complémentaire renvoyé dans le résultat de l'étape précédente que vous souhaitez créer.
   - Remplacez *version-number* par la version renvoyée dans le résultat de l'étape précédente que vous souhaitez utiliser.
   - Si le module complémentaire ne nécessite pas d'autorisations IAM, *<service-account-configuration>* supprimez-le.
   - Si le module complémentaire (1) nécessite des autorisations IAM et (2) votre cluster utilise EKS Pod Identities, remplacez-le par *<service-account-configuration>* l'association d'identité de pod suivante. *<service-account-name>*Remplacez-le par le nom du compte de service utilisé par le module complémentaire. Remplacez *<role-arn>* par l'ARN d'un rôle IAM. Le rôle doit respecter la politique de confiance requise par EKS Pod Identities.
     - `--pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'`
   - Si le module complémentaire (1) nécessite des autorisations IAM et (2) votre cluster utilise IRSA, remplacez-le *<service-account-configuration>* par la configuration IRSA suivante. *111122223333*Remplacez-le par votre identifiant de compte et *role-name* par le nom d'un rôle IAM existant que vous avez créé. Pour obtenir des instructions sur la création d'un rôle, consultez la documentation du module complémentaire que vous créez. Pour spécifier un rôle de compte de service, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
     - `--service-account-role-arn arn:aws:iam::111122223333:role/role-name`

- Cet exemple de commande remplace l'option `--configuration-values` de toute version autogérée existante du module complémentaire, le cas échéant. Remplacez cela par les valeurs de configuration souhaitées, telles qu'une chaîne ou une entrée de fichier. Si vous ne voulez pas fournir de valeurs de configuration, supprimez l'option `--configuration-values`. Si vous ne souhaitez pas AWS CLI remplacer la configuration d'un module complémentaire autogéré existant, supprimez l'*`--resolve-conflicts OVERWRITE`*option. Si vous supprimez cette option et que le module complémentaire doit remplacer la configuration d'un module complémentaire autogéré existant, la création du module complémentaire Amazon EKS échoue et vous recevez un message d'erreur pour vous aider à résoudre le conflit. Avant de spécifier cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas les paramètres que vous devez gérer, car ces paramètres sont remplacés par cette option.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    <service-account-configuration> --configuration-values '{"resources":
{"limits":{"cpu":"100m"}}}' --resolve-conflicts OVERWRITE
```

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
   <service-account-configuration> --configuration-values 'file://example.yaml'
 --resolve-conflicts OVERWRITE
```

Pour la liste complète des options disponibles, consultez `create-addon` dans la référence des lignes de commande Amazon EKS. Si le module complémentaire que vous avez créé indique `aws-marketplace` dans la colonne `Owner` d'une étape précédente, la création risque d'échouer et vous pouvez recevoir un message d'erreur similaire au suivant.

```
{
    "addon": {
        "addonName": "addon-name",
        "clusterName": "my-cluster",
        "status": "CREATE_FAILED",
        "addonVersion": "version",
        "health": {
            "issues": [
                {
                    "code": "AddonSubscriptionNeeded",
```

```
                    "message": "You are currently not subscribed to this add-
on. To subscribe, visit the AWS Marketplace console, agree to the seller EULA,
 select the pricing type if required, then re-install the add-on"
[...]
```

Si vous recevez une erreur similaire à celle de la sortie précédente, consultez l'URL figurant dans la sortie d'une étape précédente pour vous abonner au module complémentaire. Une fois abonné, exécutez à nouveau la commande `create-addon`.

Mise à jour d'un module complémentaire

Amazon EKS ne met pas automatiquement à jour un module complémentaire lorsque de nouvelles versions sont publiées ou après avoir mis à jour votre cluster vers une nouvelle version mineure de Kubernetes. Pour mettre à jour un module complémentaire pour un cluster existant, vous devez lancer la mise à jour. Ensuite, Amazon EKS met à jour le module complémentaire Amazon EKS pour vous. Avant de mettre à jour un module complémentaire, consultez la documentation relative au module complémentaire. Pour obtenir la liste des modules complémentaires disponibles, consultez Modules complémentaires Amazon EKS disponibles sur Amazon EKS. Si le module complémentaire exige un rôle IAM, consultez les détails du module complémentaire spécifique dans Modules complémentaires Amazon EKS disponibles sur Amazon EKS pour obtenir des détails sur la création du rôle.

Vous pouvez mettre à jour un module complémentaire Amazon EKS à l'aide du `eksctl` AWS Management Console, ou du AWS CLI.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour mettre à jour un module complémentaire Amazon EKS à l'aide de **`eksctl`**

1. Déterminez les modules complémentaires et les versions de modules complémentaires actuellement installés sur votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster.

```
eksctl get addon --cluster my-cluster
```

L'exemple qui suit illustre un résultat.

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0                v1.23.8-eksbuild.2
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0                v1.12.0-
eksbuild.1,v1.11.4-eksbuild.1,v1.11.3-eksbuild.1,v1.11.2-eksbuild.1,v1.11.0-
eksbuild.1
```

Votre sortie peut être différente, en fonction des modules complémentaires et des versions dont vous disposez sur votre cluster. Vous pouvez voir que dans l'exemple de sortie précédent, deux modules complémentaires existants sur le cluster disposent de versions plus récentes disponibles dans la colonne `UPDATE AVAILABLE`.

2. Mettez à jour le module complémentaire.

   1. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande :

      - Remplacez *`my-cluster`* par le nom de votre cluster.
      - *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.
      - Remplacez *`vpc-cni`* par le nom d'un module complémentaire renvoyé dans la sortie de l'étape précédente que vous souhaitez mettre à jour.
      - Si vous souhaitez procéder à une mise à jour vers une version antérieure à la dernière version disponible, remplacez *`latest`* par le numéro de version renvoyé dans la sortie de l'étape précédente que vous souhaitez utiliser. Certains modules complémentaires présentent des versions recommandées. Pour plus d'informations, consultez la documentation du module complémentaire que vous mettez à jour.
      - Si le module complémentaire utilise un compte de service Kubernetes et un rôle IAM, remplacez *`111122223333`* par votre ID de compte et *`role-name`* par le nom d'un rôle IAM existant que vous avez créé. Pour obtenir des instructions sur la création d'un rôle, consultez la documentation du module complémentaire que vous créez. Pour spécifier un rôle de compte de service, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.

Si le module complémentaire n'utilise aucun compte de service Kubernetes ni rôle IAM, supprimez la ligne **`serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name`**.

- L'*`preserve`*option préserve les valeurs existantes pour le module complémentaire. Si vous avez défini des valeurs personnalisées pour les paramètres des modules complémentaires et que vous n'utilisez pas cette option, Amazon EKS remplace vos valeurs par ses valeurs par défaut. Si vous utilisez cette option, nous vous recommandons de tester les modifications de champ et de valeur sur un cluster hors production avant de mettre à jour le module complémentaire sur votre cluster de production. Si vous remplacez cette valeur par `overwrite`, tous les paramètres sont remplacés par les valeurs par défaut d'Amazon EKS. Si vous avez défini des valeurs personnalisées pour certains paramètres, il est possible qu'elles soient remplacées par les valeurs par défaut d'Amazon EKS. Si vous remplacez cette valeur par `none`, Amazon EKS ne modifie la valeur d'aucun paramètre, mais la mise à jour risque d'échouer. Si la mise à jour échoue, vous recevez un message d'erreur pour vous aider à résoudre le conflit.

```
cat >update-addon.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code

addons:
- name: vpc-cni
  version: latest
  serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name
  resolveConflicts: preserve
EOF
```

2. Exécutez la commande modifiée pour créer le fichier `update-addon.yaml`.
3. Appliquez le fichier de configuration à votre cluster.

```
eksctl update addon -f update-addon.yaml
```

Pour plus d'informations sur la mise à jour des modules complémentaires, consultez Modules complémentaires dans la documentation `eksctl`.

AWS Management Console

Pour mettre à jour un module complémentaire Amazon EKS à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire.
3. Choisissez l'onglet Modules complémentaires.
4. Cochez la case en haut à droite de la zone du module complémentaire, puis sélectionnez Edit (Modifier).
5. Sur la page Configure ***nom du module complémentaire*** (Configurer) :
   - Sélectionnez la version que vous souhaitez utiliser. Le module complémentaire peut présenter une version recommandée. Pour plus d'informations, consultez la documentation du module complémentaire que vous mettez à jour.
   - Pour le rôle Select IAM, vous pouvez utiliser le rôle IAM du nœud (Non défini) ou un rôle existant que vous avez créé pour être utilisé avec le module complémentaire. S'il n'y a aucun rôle à sélectionner, vous n'avez pas de rôle existant. Quelle que soit l'option choisie, consultez la documentation du module complémentaire que vous créez pour créer une politique IAM et l'associer à un rôle. Pour sélectionner un rôle IAM, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.
   - Pour `Code editor`, entrez toutes les informations de configuration spécifiques au module complémentaire. Pour plus d'informations, consultez la documentation du module complémentaire que vous mettez à jour.
   - Pour Conflict resolution method (Méthode de résolution des conflits), sélectionnez l'une des options. Si vous avez défini des valeurs personnalisées pour les paramètres des modules complémentaires, nous vous recommandons l'option Preserve (Conserver). Si vous

ne choisissez pas cette option, Amazon EKS remplace vos valeurs par ses valeurs par défaut. Si vous utilisez cette option, nous vous recommandons de tester les modifications de champ et de valeur sur un cluster hors production avant de mettre à jour le module complémentaire sur votre cluster de production.

6. Choisissez Mettre à jour.

AWS CLI

Prérequis

Version `2.12.3` ou version ultérieure `1.27.160` ou version ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.

Pour mettre à jour un module complémentaire Amazon EKS à l'aide du AWS CLI

1. Consultez la liste des modules complémentaires installés. Remplacez *`my-cluster`* par le nom de votre cluster.

```
aws eks list-addons --cluster-name my-cluster
```

L'exemple qui suit illustre un résultat.

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni"
    ]
}
```

2. Affichez la version actuelle du module complémentaire que vous souhaitez mettre à jour. Remplacez *my-cluster* par le nom de votre cluster et *vpc-cni* par le nom du module complémentaire que vous souhaitez mettre à jour.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

L'exemple qui suit illustre un résultat.

```
v1.10.4-eksbuild.1
```

3. Vous pouvez voir les versions du module complémentaire disponibles pour la version de votre cluster. Remplacez *1.30* par la version de votre cluster et *vpc-cni* par le nom du module complémentaire que vous souhaitez mettre à jour.

```
aws eks describe-addon-versions --kubernetes-version 1.30 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

L'exemple qui suit illustre un résultat.

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+---------------------+
| Defaultversion  |       Version       |
+-----------------+---------------------+
|  False          |  v1.12.0-eksbuild.1 |
|  True           |  v1.11.4-eksbuild.1 |
|  False          |  v1.10.4-eksbuild.1 |
|  False          |  v1.9.3-eksbuild.1  |
+-----------------+---------------------+
```

La version avec `True` dans la colonne `Defaultversion` correspond à la version avec laquelle le module complémentaire est créé, par défaut.

4. Mettez à jour votre module complémentaire. Copiez la commande qui suit sur votre appareil. Si nécessaire, apportez les modifications suivantes à la commande, puis exécutez la commande modifiée.

   - Remplacez *my-cluster* par le nom de votre cluster.

- Remplacez *vpc-cni* par le nom du module complémentaire que vous souhaitez mettre à jour renvoyé dans la sortie d'une étape précédente.
- Remplacez *version-number* par la version renvoyée dans la sortie de l'étape précédente vers laquelle vous souhaitez effectuer la mise à jour. Certains modules complémentaires présentent des versions recommandées. Pour plus d'informations, consultez la documentation du module complémentaire que vous mettez à jour.
- Si le module complémentaire utilise un compte de service Kubernetes et un rôle IAM, remplacez *111122223333* par votre ID de compte et *role-name* par le nom d'un rôle IAM existant que vous avez créé. Pour obtenir des instructions sur la création d'un rôle, consultez la documentation du module complémentaire que vous créez. Pour spécifier un rôle de compte de service, vous devez disposer d'un fournisseur IAM OpenID Connect (OIDC) pour votre cluster. Pour déterminer si vous en avez un ou pour en créer un, consultez Créez un OIDC fournisseur IAM pour votre cluster.

  Si le module complémentaire n'utilise aucun compte de service Kubernetes ni rôle IAM, supprimez la ligne **serviceAccountRoleARN: arn:aws:iam::*111122223333*:role/*role-name***.
- L'option **--resolve-conflicts** *PRESERVE* (CONSERVER) conserve les valeurs existantes pour le module complémentaire. Si vous avez défini des valeurs personnalisées pour les paramètres des modules complémentaires et que vous n'utilisez pas cette option, Amazon EKS remplace vos valeurs par ses valeurs par défaut. Si vous utilisez cette option, nous vous recommandons de tester les modifications de champ et de valeur sur un cluster hors production avant de mettre à jour le module complémentaire sur votre cluster de production. Si vous remplacez cette valeur par `overwrite`, tous les paramètres sont remplacés par les valeurs par défaut d'Amazon EKS. Si vous avez défini des valeurs personnalisées pour certains paramètres, il est possible qu'elles soient remplacées par les valeurs par défaut d'Amazon EKS. Si vous remplacez cette valeur par `none`, Amazon EKS ne modifie la valeur d'aucun paramètre, mais la mise à jour risque d'échouer. Si la mise à jour échoue, vous recevez un message d'erreur pour vous aider à résoudre le conflit.
- Si vous voulez supprimer toutes les configurations personnalisées, effectuez la mise à jour à l'aide de l'option *--configuration-values '{}'*. Cela ramène toutes les configurations personnalisées aux valeurs par défaut. Si vous ne voulez pas modifier votre configuration personnalisée, ne fournissez pas l'indicateur *--configuration-values*. Si vous voulez ajuster une configuration personnalisée, remplacez *{}* par les nouveaux

paramètres. Pour consulter la liste des paramètres, consultez l'étape Affichage du schéma de configuration dans la section Création d'un module complémentaire.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
configuration-values '{}' --resolve-conflicts PRESERVE
```

5. Vérifiez l'état de la mise à jour. Remplacez *my-cluster* par le nom de votre cluster et *vpc-cni* par le nom du module complémentaire que mettez à jour.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

L'exemple qui suit illustre un résultat.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "UPDATING",
[...]
```

Lorsque la mise à jour est terminé, l'état indique `ACTIVE`.

Suppression d'un module complémentaire

Lorsque vous supprimez un module complémentaire Amazon EKS :

- Il n'y a pas de temps d'arrêt pour les fonctionnalités fournies par le module complémentaire.
- Si vous utilisez des rôles IAM pour les comptes de service (IRSA) et qu'un rôle IAM est associé au module complémentaire, celui-ci n'est pas supprimé.
- Si vous utilisez Pod Identities, toutes les associations Pod Identity détenues par le module complémentaire sont supprimées. Si vous spécifiez l'`--preserve`option sur AWS CLI, les associations sont préservées.
- Amazon EKS cesse de gérer les paramètres du module complémentaire.
- La console cesse de vous avertir lorsque de nouvelles versions sont disponibles.
- Vous ne pouvez pas mettre à jour le module complémentaire à l'aide d' AWS outils ou d'API.

- Vous pouvez choisir de laisser le module complémentaire sur votre cluster afin de pouvoir le gérer automatiquement, ou le supprimer de votre cluster. Vous ne devez supprimer le module complémentaire de votre cluster que si aucune ressource sur votre cluster ne dépend de la fonctionnalité fournie par ce module complémentaire.

Vous pouvez supprimer un module complémentaire Amazon EKS de votre cluster à l'aide de `eksctl`, de la AWS Management Console ou de la AWS CLI.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour supprimer un module complémentaire Amazon EKS à l'aide de **`eksctl`**

1. Déterminez les modules complémentaires actuellement installés sur votre cluster. Remplacez *`my-cluster`* par le nom de votre cluster.

```
eksctl get addon --cluster my-cluster
```

L'exemple qui suit illustre un résultat.

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0
[...]
```

Votre sortie peut être différente, en fonction des modules complémentaires et des versions dont vous disposez sur votre cluster.

2. Supprimez le module complémentaire. Remplacez *`my-cluster`* par le nom de votre cluster et *`name-of-add-on`* par le nom du module complémentaire renvoyé dans la sortie de l'étape précédente que vous souhaitez supprimer. Si vous supprimez l'option ***`--preserve`***, Amazon EKS ne gère plus le module complémentaire et ce dernier est supprimé de votre cluster.

```
eksctl delete addon --cluster my-cluster --name name-of-addon --preserve
```

AWS Management Console

Pour supprimer un module complémentaire Amazon EKS à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, sélectionnez Clusters, puis sélectionnez le nom du cluster pour lequel vous souhaitez supprimer le module complémentaire Amazon EKS.
3. Choisissez l'onglet Modules complémentaires.
4. Cochez la case en haut à droite de la zone du module complémentaire, puis choisissez Remove (Supprimer). Sélectionnez Preserve on the cluster (Préserver sur un cluster) si vous voulez qu'Amazon EKS cesse de gérer les paramètres du module complémentaire tout en souhaitant retenir le module complémentaire sur votre cluster afin de pouvoir gérer automatiquement tous ses paramètres. Saisissez le nom du module complémentaire, puis sélectionnez Remove (Supprimer).

AWS CLI

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour supprimer un module complémentaire Amazon EKS à l'aide du AWS CLI

1. Consultez la liste des modules complémentaires installés. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks list-addons --cluster-name my-cluster
```

L'exemple qui suit illustre un résultat.

```
{
```

```
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni",
        "name-of-addon"
    ]
}
```

2. Supprimez le module complémentaire installé. Remplacez *my-cluster* par le nom de votre cluster et *name-of-add-on* par le nom du module complémentaire que vous souhaitez supprimer. La suppression de ***--preserve*** supprime le module complémentaire de votre cluster.

```
aws eks delete-addon --cluster-name my-cluster --addon-name name-of-addon --
preserve
```

L'exemple suivant illustre le résultat abrégé.

```
{
    "addon": {
        "addonName": "name-of-add-on",
        "clusterName": "my-cluster",
        "status": "DELETING",
[...]
```

3. Vérifier l'état de la suppression. Remplacez *my-cluster* par le nom de votre cluster et *name-of-addon* par le nom du module complémentaire que supprimez.

```
aws eks describe-addon --cluster-name my-cluster --addon-name name-of-addon
```

L'exemple suivant illustre le résultat après la suppression du module complémentaire.

```
An error occurred (ResourceNotFoundException) when calling the DescribeAddon
 operation: No addon: name-of-addon found in cluster: my-cluster
```

Récupérez la compatibilité des versions de l'addon

Utilisez l'describe-addon-verisionsAPI pour répertorier les versions disponibles des addons EKS et les versions de Kubernetes prises en charge par chaque version d'addon.

Récupérer la compatibilité des versions de l'addon ()AWS CLI

1. Vérifiez que le AWS CLI est installé et fonctionne avec`aws sts get-caller-identity`. Si cette commande ne fonctionne pas, découvrez comment démarrer avec le AWS CLI.
2. Déterminez le nom de l'extension pour laquelle vous souhaitez récupérer les informations de compatibilité des versions, par exemple`amazon-cloudwatch-observability`.
3. Déterminez la version Kubernetes de votre cluster, par exemple. `1.28`
4. Utilisez le AWS CLI pour récupérer les versions d'addon compatibles avec la version Kubernetes de votre cluster.

```
aws eks describe-addon-versions --addon-name amazon-cloudwatch-observability --
kubernetes-version 1.29
```

L'exemple qui suit illustre un résultat.

```
{
    "addons": [
        {
            "addonName": "amazon-cloudwatch-observability",
            "type": "observability",
            "addonVersions": [
                {
                    "addonVersion": "v1.5.0-eksbuild.1",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.28",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
[...]
```

Cette sortie indique que la version de l'addon `v1.5.0-eksbuild.1` est compatible avec la version du cluster Kubernetes. `1.28`

Gestion des champs Kubernetes

Les modules complémentaires Amazon EKS sont installés sur votre cluster à l'aide des configurations standard de bonnes pratiques. Pour plus d'informations sur l'ajout d'un module complémentaire Amazon EKS à votre cluster, consultez Modules complémentaires Amazon EKS.

Vous voudrez peut-être personnaliser la configuration d'un module complémentaire Amazon EKS pour activer les fonctions avancées. Amazon EKS utilise la fonction d'application côté serveur Kubernetes pour permettre la gestion d'un module complémentaire par Amazon EKS sans écraser votre configuration pour les paramètres qui ne sont pas gérés par Amazon EKS. Pour plus d'informations, consultez Server-Side Apply dans la documentation Kubernetes. Pour ce faire, Amazon EKS gère un ensemble minimal de champs pour chaque module complémentaire qu'il installe. Vous pouvez, sans problème, modifier tous les champs qui ne sont pas gérés par Amazon EKS ou un autre processus de plan de contrôle Kubernetes tel que `kube-controller-manager`.

> **Important**
>
> La modification d'un champ géré par Amazon EKS empêche Amazon EKS de gérer le module complémentaire. Ce processus peut entraîner l'écrasement de vos modifications lors de la mise à jour d'un module complémentaire.

Afficher l'état de gestion des champs

Vous pouvez utiliser `kubectl` pour voir quels champs sont gérés par Amazon EKS pour tout module complémentaire Amazon EKS.

Pour voir l'état de gestion d'un champ

1. Déterminez le module complémentaire que vous souhaitez examiner. Pour afficher tous les `deployments` et `DaemonSets` déployés sur votre cluster, consultez Afficher les ressources Kubernetes.
2. Pour afficher les champs gérés d'un module complémentaire, exécutez la commande suivante :

   ```
   kubectl get type/add-on-name -n add-on-namespace -o yaml
   ```

   Par exemple, vous pouvez voir les champs gérés pour le module complémentaire CoreDNS avec la commande suivante.

```
kubectl get deployment/coredns -n kube-system -o yaml
```

La gestion des champs est répertoriée dans la section suivante dans la sortie renvoyée.

```
[...]
managedFields:
  - apiVersion: apps/v1
    fieldsType: FieldsV1
    fieldsV1:
[...]
```

> **Note**
>
> Si vous ne voyez pas `managedFields` en sortie, ajoutez **`--show-managed-fields`** à la commande et exécutez-la à nouveau. La version de `kubectl` que vous utilisez détermine si les champs gérés sont retournés par défaut.

Présentation de la syntaxe de gestion des champs dans l'API Kubernetes

Lorsque vous affichez les détails d'un objet Kubernetes, les champs gérés et non gérés sont renvoyés dans la sortie. Les champs gérés peuvent être l'un des types suivants :

- Entièrement géré : toutes les clés du champ sont gérées par Amazon EKS. Les modifications apportées à une valeur provoquent un conflit.
- Partiellement géré : certaines clés du champ sont gérées par Amazon EKS. Seules les modifications apportées aux clés explicitement gérées par Amazon EKS provoquent un conflit.

Les deux types de champs sont étiquetés avec `manager: eks`.

Chaque clé est soit un `.` représentant le champ lui-même, qui correspond toujours à un ensemble vide, soit une chaîne qui représente un sous-champ ou un élément. La sortie pour la gestion des champs comprend les types de déclarations suivants :

- `f:`*`name`*, où *`name`* est le nom d'un champ d'une liste.
- `k:`*`keys`*, où *`keys`* est une carte des champs de l'élément d'une liste.
- `v:`*`value`*, où *`value`* est la valeur exacte au format JSON formaté de l'élément d'une liste.

- `i:index`, où *`index`* est la position d'un élément dans la liste.

Les parties de sortie suivantes pour le module complémentaire CoreDNS illustrent les déclarations précédentes :

- Champs entièrement gérés : si un champ géré possède un (champ) `f:` spécifié, mais aucune (clé) `k:`, alors le champ entier est géré. Les modifications apportées aux valeurs de ce champ provoquent un conflit.

  Dans la sortie suivante, vous pouvez voir que le conteneur nommé `coredns` est géré par `eks`. Les sous-champs `args`, `image` et `imagePullPolicy` sont également gérés par `eks`. Les modifications apportées aux valeurs de ces champs provoquent un conflit.

  ```
  [...]
  f:containers:
    k:{"name":"coredns"}:
    .: {}
    f:args: {}
    f:image: {}
    f:imagePullPolicy: {}
  [...]
  manager: eks
  [...]
  ```

- Champs partiellement gérés : si une clé gérée a une valeur spécifiée, les clés déclarées sont gérées pour ce champ. La modification des clés spécifiées provoque un conflit.

  Dans la sortie suivante, vous pouvez voir que `eks` gère les volumes `config-volume` et `tmp` définis à l'aide de la clé `name`.

  ```
  [...]
  f:volumes:
    k:{"name":"config-volume"}:
      .: {}
      f:configMap:
        f:items: {}
        f:name: {}
      f:name: {}
    k:{"name":"tmp"}:
      .: {}
      f:name: {}
  ```

```
[...]
manager: eks
[...]
```

- Ajout de clés à des champs partiellement gérés : si seule une valeur de clé spécifique est gérée, vous pouvez ajouter en toute sécurité des clés supplémentaires, telles que des arguments, à un champ sans provoquer de conflit. Si vous ajoutez des clés supplémentaires, assurez-vous d'abord que le champ n'est pas géré. L'ajout ou la modification d'une valeur gérée provoque un conflit.

  Dans la sortie suivante, vous pouvez voir que la clé `name` et le champ `name` sont gérés. L'ajout ou la modification d'un nom de conteneur provoque un conflit avec cette clé gérée.

```
[...]
f:containers:
  k:{"name":"coredns"}:
[...]
    f:name: {}
[...]
manager: eks
[...]
```

Associer un rôle IAM à un module complémentaire Amazon EKS à l'aide de Pod Identity

Certains modules complémentaires Amazon EKS nécessitent des autorisations de rôle IAM pour appeler des AWS API. Par exemple, le module complémentaire Amazon VPC CNI appelle certaines AWS API pour configurer les ressources réseau de votre compte. Ces modules complémentaires doivent être autorisés à l'aide d' AWS IAM. Plus précisément, le compte de service du pod exécutant le module complémentaire doit être associé à un rôle IAM doté d'une politique IAM suffisante.

La méthode recommandée pour accorder des AWS autorisations aux charges de travail de cluster est d'utiliser la fonctionnalité Pod Identities d'Amazon EKS. Vous pouvez utiliser une association d'identité Pod pour associer le compte de service d'un module complémentaire à un rôle IAM.
Si un pod utilise un compte de service qui a une association, Amazon EKS définit des variables d'environnement dans les conteneurs du pod. Les variables d'environnement configurent les AWS SDK, y compris la AWS CLI, pour utiliser les informations d'identification EKS Pod Identity. En savoir plus sur EKS Pod Identities.

Les modules complémentaires Amazon EKS peuvent aider à gérer le cycle de vie des associations d'identité des pods correspondant au module complémentaire. Par exemple, vous pouvez créer ou mettre à jour un module complémentaire Amazon EKS et l'association d'identité de pod nécessaire en un seul appel d'API. Amazon EKS fournit également une API permettant de récupérer les politiques IAM suggérées.

Utilisation suggérée :

1. Vérifiez que l'agent d'identité du pod Amazon EKS est configuré sur votre cluster.
2. Déterminez si le module complémentaire que vous souhaitez installer nécessite des autorisations IAM à l'aide de l'`describe-addon-versions` AWS CLI opération. Si l'`requiresIamPermissions`indicateur l'est`true`, vous devez utiliser l'`describe-addon-configuration`opération pour déterminer les autorisations requises par l'addon. La réponse inclut une liste de stratégies IAM gérées suggérées.
3. Récupérez le nom du compte de service Kubernetes et la politique IAM suggérée à l'aide de l'opération CLI. `describe-addon-configuration` Évaluez la portée de la politique suggérée par rapport à vos exigences de sécurité.
4. Créez un rôle IAM en utilisant la politique d'autorisation suggérée et la politique de confiance requise par Pod Identity. Pour plus d'informations, consultez Création de l'association d'identité du pod EKS.
5. Créez ou mettez à jour un module complémentaire Amazon EKS à l'aide de la CLI. Spécifiez au moins une association d'identité de pod. Une association d'identité de pod est (1) le nom d'un compte de service Kubernetes et (2) l'ARN d'un rôle IAM.

Considérations :

- Les associations d'identité de pod créées à l'aide des API du module complémentaire appartiennent au module complémentaire correspondant. Si vous supprimez le module complémentaire, l'association d'identité du module est également supprimée. Vous pouvez empêcher cette suppression en cascade en utilisant l'`preserve`option lors de la suppression d'un addon à l'aide de l' AWS CLI API or. Vous pouvez également mettre à jour ou supprimer directement l'association d'identité du module si nécessaire. Les modules complémentaires ne peuvent pas assumer la propriété des associations d'identité de pods existantes. Vous devez supprimer l'association existante et la recréer à l'aide d'une opération de création ou de mise à jour d'un module complémentaire.

- Amazon EKS recommande d'utiliser les associations d'identité des pods pour gérer les autorisations IAM pour les modules complémentaires. La méthode précédente, les rôles IAM pour les comptes de service (IRSA), est toujours prise en charge. Vous pouvez spécifier à la fois une association d'identité IRSA `serviceAccountRoleArn` et une association d'identité de pod pour un module complémentaire. Si l'agent d'identité du pod EKS est installé sur le cluster, il `serviceAccountRoleArn` sera ignoré et EKS utilisera l'association d'identité du pod fournie. Si Pod Identity n'est pas activé, il `serviceAccountRoleArn` sera utilisé.
- Si vous mettez à jour les associations d'identité des modules pour un module complémentaire existant, Amazon EKS lance un redémarrage progressif des modules complémentaires.

Récupérer les informations IAM relatives à un module complémentaire

Vous pouvez utiliser le AWS CLI pour déterminer (1) si un module complémentaire nécessite des autorisations IAM, et (2) une stratégie IAM suggérée pour ce module complémentaire.

Récupérer les informations IAM relatives à un module complémentaire Amazon EKS ()AWS CLI

1. Déterminez le nom du module complémentaire que vous souhaitez installer et la version Kubernetes de votre cluster. En savoir plus sur les modules complémentaires Amazon EKS disponibles.
2. Utilisez le AWS CLI pour déterminer si le module complémentaire nécessite des autorisations IAM.

   ```
   aws eks describe-addon-versions \
   --addon-name <addon-name> \
   --kubernetes-version <kubernetes-version>
   ```

   Par exemple :

   ```
   aws eks describe-addon-versions \
   --addon-name aws-ebs-csi-driver \
   --kubernetes-version 1.30
   ```

   Passez en revue l'exemple de sortie suivant. Notez que `requiresIamPermissions` c'est `true` la version du module complémentaire par défaut. Vous devez spécifier la version du module complémentaire lors de la récupération de la politique IAM recommandée.

```
{
    "addons": [
        {
            "addonName": "aws-ebs-csi-driver",
            "type": "storage",
            "addonVersions": [
                {
                    "addonVersion": "v1.31.0-eksbuild.1",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.30",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
                    "requiresConfiguration": false,
                    "requiresIamPermissions": true
                },
[...]
```

3. Si le module complémentaire nécessite des autorisations IAM, utilisez le AWS CLI pour récupérer une politique IAM recommandée.

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name <addon-name> \
--addon-version <addon-version>
```

Par exemple :

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name aws-ebs-csi-driver \
--addon-version v1.31.0-eksbuild.1
```

Passez en revue le résultat suivant. Notez le `recommendedManagedPolicies`.

```
[
    {
        "serviceAccount": "ebs-csi-controller-sa",
        "recommendedManagedPolicies": [
            "arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy"
        ]
    }
]
```

4. Créez un rôle IAM et associez la politique gérée recommandée. Vous pouvez également passer en revue la politique gérée et limiter les autorisations le cas échéant. Consultez les instructions relatives à la création d'un rôle IAM à utiliser avec EKS Pod Identities.

Mettre à jour le module complémentaire avec le rôle IAM

Mettre à jour un module complémentaire Amazon EKS pour utiliser une association d'identité Pod (AWS CLI)

1. Déterminez :

   - `cluster-name`— Le nom du cluster EKS sur lequel installer le module complémentaire.
   - `addon-name`— Le nom du module complémentaire Amazon EKS à installer.
   - `service-account-name`— Le nom du compte de service Kubernetes utilisé par le module complémentaire.
   - `iam-role-arn`— L'ARN d'un rôle IAM doté d'autorisations suffisantes pour le module complémentaire. Le rôle IAM doit disposer de la politique de confiance requise pour EKS Pod Identity.

2. Mettez à jour le module complémentaire à l'aide de la AWS CLI. Vous pouvez également spécifier les associations d'identité des pods lors de la création d'un module complémentaire, en utilisant la même `--pod-identity-assocations` syntaxe. Notez que lorsque vous spécifiez des associations d'identité d'espace lors de la mise à jour d'un module complémentaire, toutes les associations d'identité d'espace précédentes sont remplacées.

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
```

```
--pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
```

Par exemple :

```
aws eks update-addon --cluster-name mycluster \
--addon-name aws-ebs-csi-driver \
--pod-identity-associations 'serviceAccount=ebs-csi-controller-sa,roleArn=arn:aws:iam::123456789012:role/StorageDriver'
```

3. Vérifiez que l'association d'identité du pod a été créée :

```
aws eks list-pod-identity-associations --cluster-name <cluster-name>
```

Si la commande aboutit, vous devriez obtenir une sortie similaire à ce qui suit. Notez le OwnerArn du module complémentaire EKS.

```
{
    "associations": [
        {
            "clusterName": "mycluster",
            "namespace": "kube-system",
            "serviceAccount": "ebs-csi-controller-sa",
            "associationArn": "arn:aws:eks:us-west-2:123456789012:podidentityassociation/mycluster/a-4wvljrezsukshq1bv",
            "associationId": "a-4wvljrezsukshq1bv",
            "ownerArn": "arn:aws:eks:us-west-2:123456789012:addon/mycluster/aws-ebs-csi-driver/9cc7ce8c-2e15-b0a7-f311-426691cd8546"
        }
    ]
}
```

Supprimer les associations du module complémentaire

Supprimer toutes les associations d'identité des pods d'un module complémentaire Amazon EKS (AWS CLI)

1. Déterminez :

   - `cluster-name`— Le nom du cluster EKS sur lequel installer le module complémentaire.

- `addon-name`— Le nom du module complémentaire Amazon EKS à installer.

2. Mettez à jour l'addon pour spécifier un tableau vide d'associations d'identité de pod.

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
--pod-identity-associations "[]"
```

Résoudre les problèmes liés aux identités des modules pour les modules complémentaires EKS

Si vos modules complémentaires rencontrent des erreurs lors de leurs tentatives d' AWS utilisation de l'API, du SDK ou de la CLI, vérifiez les points suivants :

- L'agent Pod Identity est installé dans votre cluster.
  - Découvrez comment configurer l'agent Pod Identity.
- Le module complémentaire possède une association d'identité de pod valide.
  - Utilisez le AWS CLI pour récupérer les associations pour le nom du compte de service utilisé par le module complémentaire.

```
aws eks list-pod-identity-associations --cluster-name <cluster-name>
```

- Le rôle IAM prévu possède la politique de confiance requise pour EKS Pod Identities.
  - Utilisez le AWS CLI pour récupérer la politique de confiance d'un module complémentaire.

```
aws iam get-role --role-name <role-name> --query Role.AssumeRolePolicyDocument
```

- Le rôle IAM prévu dispose des autorisations nécessaires pour le module complémentaire.
  - AWS CloudTrail À utiliser pour passer en revue `AccessDenied` des `UnauthorizedOperation` événements.
- Le nom du compte de service indiqué dans l'association d'identité du module correspond au nom du compte de service utilisé par le module complémentaire.
  - Consultez la documentation du module complémentaire pour déterminer le nom du compte de service.

Vérification d'une image de conteneur lors du déploiement

Si vous utilisez AWS Signer et souhaitez vérifier des images de conteneur signées au moment du déploiement, vous pouvez utiliser l'une des solutions suivantes :

- Gatekeeper et Ratify : utilisez Gatekeeper comme contrôleur d'admission et Ratify est configuré avec un plug-in AWS Signer en tant que hook Web pour valider les signatures.
- Kyverno — Un moteur de politiques Kubernetes configuré avec un plugin AWS Signer pour valider les signatures.

> **Note**
>
> Avant de vérifier les signatures des images des conteneurs, configurez le référentiel de confiance Notation et la politique de confiance, comme l'exige le contrôleur d'admission que vous avez sélectionné.

Formation au machine learning avec Elastic Fabric Adapter

Cette rubrique décrit l'intégration d'Elastic Fabric Adapter (EFA) aux Pods déployés dans votre cluster Amazon EKS. Elastic Fabric Adapter (EFA) est une interface réseau pour les instances Amazon EC2 qui vous permet d'exécuter des applications nécessitant des niveaux élevés de communications inter-nœuds à grande échelle sur AWS. Son interface matérielle sur mesure de contournement du système d'exploitation améliore les performances des communications entre instances, ce qui est essentiel à la mise à l'échelle de ces applications. Avec EFA, les applications de calcul haute performance (HPC) utilisant MPI (Message Passing Interface) et les applications de ML (Machine Learning) utilisant NVIDIA Collective Communications Library (NCCL) peuvent avoir une capacité de mise à l'échelle pouvant atteindre des milliers de CPU ou de GPU. Vous bénéficiez ainsi des performances applicatives des clusters HPC sur site avec l'élasticité et la flexibilité à la demande du AWS cloud. L'intégration d'EFA aux applications exécutées sur des clusters Amazon EKS permet de réduire le temps nécessaire à l'exécution des applications d'entraînement distribué à grande échelle sans avoir à ajouter d'autres instances à votre cluster. Pour plus d'informations sur l'EFA, consultez Elastic Fabric Adapter.

Le plugin EFA décrit dans cette rubrique prend entièrement en charge les instances Amazon EC2 P4d, qui représentent l'état actuel de l'art en matière de machine learning distribué dans le cloud. Chaque instance `p4d.24xlarge` possède huit GPU NVIDIA A100 avec GPUDirectRDMA

400 Gbit/s sur EFA. GPUDirectRDMA vous permet d'avoir une communication directe de GPU à GPU entre les nœuds avec un contournement du processeur, ce qui augmente la bande passante de communication collective et réduit la latence. L'intégration d'Amazon EKS et d'EFA avec les instances P4d fournit une méthode fluide pour tirer parti de l'instance de calcul Amazon EC2 la plus performante pour l'entraînement au machine learning distribué.

Prérequis

- Un cluster Amazon EKS existant . Si vous n'avez pas de cluster existant, utilisez l'un de nos guides Démarrer avec Amazon EKS pour en créer un. Votre cluster doit être déployé dans un VPC disposant d'au moins un sous-réseau privé ayant suffisamment d'adresses IP disponibles dans lequel déployer des nœuds. Le sous-réseau privé doit disposer d'un accès Internet sortant fourni par un appareil externe, tel qu'une passerelle NAT.

  Si vous prévoyez d'utiliser `eksctl` pour créer votre groupe de nœuds, `eksctl` peut également créer un cluster pour vous.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Vous devez disposer de la version `1.7.10` de l'Amazon VPC CNI plugin for Kubernetes avant de lancer des composants master qui prennent en charge plusieurs Elastic Fabric Adapters, tels que le `p4d.24xlarge`. Pour plus d'informations sur la mise à jour de votre version Amazon VPC CNI plugin for Kubernetes, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS.

Créer un groupe de nœuds

La procédure suivante vous aide à créer un groupe de nœuds avec un groupe de nœuds soutenu `p4d.24xlarge` en utilisant des interfaces EFA et GPUDirect RDMA. Elle permet aussi d'exécuter un exemple de test NVIDIA Collective Communications Library (NCCL) pour les performances NCCL multi-nœuds à l'aide des EFA. Cet exemple peut être utilisé comme modèle d'entraînement de deep learning distribué sur Amazon EKS à l'aide des EFA.

1. Déterminez quels types d'instances Amazon EC2 prenant en charge l'EFA sont disponibles dans l'instance dans Région AWS laquelle vous souhaitez déployer des nœuds. *region-code*Remplacez-le par Région AWS celui dans lequel vous souhaitez déployer votre groupe de nœuds.

   ```
   aws ec2 describe-instance-types --region region-code --filters Name=network-
   info.efa-supported,Values=true \
       --query "InstanceTypes[*].[InstanceType]" --output text
   ```

   Lorsque vous déployez des nœuds, le type d'instance que vous souhaitez déployer doit être disponible dans l'environnement dans Région AWS lequel se trouve votre cluster.

2. Déterminez dans quelles zones de disponibilité le type d'instance que vous souhaitez déployer est disponible. Dans ce didacticiel, le type d'`p4d.24xlarge`instance est utilisé et doit être renvoyé dans la sortie pour Région AWS celui que vous avez spécifié à l'étape précédente. Lorsque vous déployez des nœuds dans un cluster de production, remplacez-les *p4d.24xlarge* par n'importe quel type d'instance renvoyé à l'étape précédente.

   ```
   aws ec2 describe-instance-type-offerings --region region-code --location-type
    availability-zone --filters Name=instance-type,Values=p4d.24xlarge \
       --query 'InstanceTypeOfferings[*].Location' --output text
   ```

   L'exemple qui suit illustre un résultat.

   ```
   us-west-2a    us-west-2c    us-west-2b
   ```

   Notez les zones de disponibilité renvoyées pour une utilisation ultérieure. Lorsque vous déployez des nœuds sur un cluster, votre VPC doit disposer de sous-réseaux avec des adresses IP disponibles dans l'une des zones de disponibilité renvoyées dans la sortie.

3. Créez un groupe de nœuds à l'aide de l'un `eksctl` ou de l'autre AWS CLI et AWS CloudFormation.

   eksctl

   Prérequis

   Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

   1. Copiez le contenu suivant dans un fichier nommé *`efa-cluster.yaml`*. Remplacez les *`example values`* par vos propres valeurs. Vous pouvez remplacer *`p4d.24xlarge`* par une instance différente. Dans ce cas, assurez-vous que les valeurs de `availabilityZones` sont des zones de disponibilité renvoyées pour le type d'instance à l'étape 1.

      ```
      apiVersion: eksctl.io/v1alpha5
      kind: ClusterConfig

      metadata:
        name: my-efa-cluster
        region: region-code
        version: "1.XX"

      iam:
        withOIDC: true

      availabilityZones: ["us-west-2a", "us-west-2c"]

      managedNodeGroups:
        - name: my-efa-ng
          instanceType: p4d.24xlarge
          minSize: 1
          desiredCapacity: 2
          maxSize: 3
          availabilityZones: ["us-west-2a"]
          volumeSize: 300
          privateNetworking: true
          efaEnabled: true
      ```

   2. Créez un groupe de nœuds gérés dans un cluster existant.

```
eksctl create nodegroup -f efa-cluster.yaml
```

Si vous n'avez pas de cluster existant, vous pouvez exécuter la commande suivante pour créer un cluster et le groupe de nœuds.

```
eksctl create cluster -f efa-cluster.yaml
```

> **Note**
>
> Étant donné que le type d'instance utilisé dans cet exemple comporte des GPU, `eksctl` installe automatiquement le plugin d'appareil NVIDIA Kubernetes sur chaque instance pour vous.

AWS CLI and AWS CloudFormation

Il existe plusieurs exigences pour la mise en réseau d'EFA, notamment la création d'un groupe de sécurité spécifique à l'EFA, la création d'un groupe de placement Amazon EC2 et la création d'un modèle de lancement qui spécifie une ou plusieurs interfaces EFA et inclut l'installation du pilote EFA dans le cadre des données utilisateur Amazon EC2. Pour en savoir plus sur les exigences relatives à l'EFA, consultez Commencer à utiliser l'EFA et le MPI dans le guide de l'utilisateur Amazon EC2. Les étapes suivantes créent tout cela pour vous. Remplacez tous les *exemples de valeurs* par les vôtres.

1. Définissez quelques variables utilisées dans des étapes ultérieures. Remplacez tous les *example values* par les vôtres. Remplacez *my-cluster* par le nom de votre cluster existant. La valeur pour `node_group_resources_name` est ensuite utilisée pour créer une AWS CloudFormation pile. La valeur de `node_group_name` est utilisée par la suite pour créer le groupe de nœuds dans votre cluster.

   ```
   cluster_name="my-cluster"
   cluster_region="region-code"
   node_group_resources_name="my-efa-nodegroup-resources"
   node_group_name="my-efa-nodegroup"
   ```

2. Identifiez un sous-réseau privé dans votre VPC qui se trouve dans la zone de disponibilité où le type d'instance que vous souhaitez déployer est disponible.

a. Récupérez la version de votre cluster et stockez-la dans une variable pour une utilisation ultérieure.

```
cluster_version=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.version" \
    --output text)
```

b. Récupérez l'ID VPC dans lequel se trouve votre cluster et stockez-le dans une variable pour une utilisation ultérieure.

```
vpc_id=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.resourcesVpcConfig.vpcId" \
    --output text)
```

c. Récupérez l'ID du groupe de sécurité de plan de contrôle pour votre cluster et stockez-le dans une variable pour une utilisation ultérieure.

```
control_plane_security_group=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.resourcesVpcConfig.clusterSecurityGroupId" \
    --output text)
```

d. Obtenez la liste des ID de sous-réseaux dans votre VPC qui se trouvent dans une zone de disponibilité renvoyée à l'étape 1.

```
aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$vpc_id" "Name=availability-
zone,Values=us-west-2a" \
    --query 'Subnets[*].SubnetId' \
    --output text
```

Si aucune sortie n'est renvoyée, essayez une autre zone de disponibilité renvoyée à l'étape 1. Si aucun de vos sous-réseaux ne se trouve dans une zone de disponibilité renvoyée à l'étape 1, vous devez créer un sous-réseau dans une zone de disponibilité renvoyée à l'étape 1. Si vous ne disposez pas d'espace dans votre VPC pour créer un autre sous-réseau, vous pouvez ajouter un bloc CIDR au VPC et créer des sous-réseaux dans le nouveau bloc CIDR, ou créer un nouveau cluster dans un nouveau VPC.

e. Déterminez si le sous-réseau est un sous-réseau privé en vérifiant la table de routage du sous-réseau.

```
aws ec2 describe-route-tables \
    --filter Name=association.subnet-id,Values=subnet-0d403852a65210a29 \
    --query "RouteTables[].Routes[].GatewayId" \
    --output text
```

L'exemple qui suit illustre un résultat.

```
local
```

Si la sortie est `local igw-02adc64c1b72722e2`, le sous-réseau est un sous-réseau public. Vous devez sélectionner un sous-réseau privé dans une zone de disponibilité renvoyée à l'étape 1. Une fois que vous avez identifié un sous-réseau privé, notez son ID pour une utilisation ultérieure.

f. Définissez une variable avec l'ID de sous-réseau privé de l'étape précédente pour une utilisation ultérieure.

```
subnet_id=your-subnet-id
```

3. Téléchargez le AWS CloudFormation modèle.

```
curl -O https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/
cloudformation/efa-p4d-managed-nodegroup.yaml
```

4. Copiez le texte suivant sur votre ordinateur. Remplacez *p4d.24xlarge* par un type d'instance à l'étape 1. Remplacez *subnet-0d403852a65210a29* par l'ID du sous-réseau privé identifié à l'étape 2.b.v. Remplacez *path-to-downloaded-cfn-template* par le chemin vers le `efa-p4d-managed-nodegroup.yaml` que vous avez téléchargé à l'étape précédente. Remplacez *your-public-key-name* par le nom de votre clé publique. Une fois les remplacements effectués, exécutez la commande modifiée.

```
aws cloudformation create-stack \
    --stack-name ${node_group_resources_name} \
    --capabilities CAPABILITY_IAM \
    --template-body file://path-to-downloaded-cfn-template \
    --parameters \
       ParameterKey=ClusterName,ParameterValue=${cluster_name} \
```

```
       ParameterKey=ClusterControlPlaneSecurityGroup,ParameterValue=
${control_plane_security_group} \
       ParameterKey=VpcId,ParameterValue=${vpc_id} \
       ParameterKey=SubnetId,ParameterValue=${subnet_id} \
       ParameterKey=NodeGroupName,ParameterValue=${node_group_name} \
       ParameterKey=NodeImageIdSSMParam,ParameterValue=/aws/service/eks/
optimized-ami/${cluster_version}/amazon-linux-2-gpu/recommended/image_id \
       ParameterKey=KeyName,ParameterValue=your-public-key-name \
       ParameterKey=NodeInstanceType,ParameterValue=p4d.24xlarge
```

5. Déterminez quand la pile que vous avez déployée à l'étape précédente est déployée.

```
aws cloudformation wait stack-create-complete --stack-name
 $node_group_resources_name
```

   Il n'y a pas de sortie de la commande précédente, mais votre invite du shell ne revient pas tant que la pile n'est pas créée.

6. Créez votre groupe de nœuds à l'aide des ressources créées par la pile AWS CloudFormation à l'étape précédente.

   a. Récupérez les informations de la AWS CloudFormation pile déployée et stockez-les dans des variables.

```
node_instance_role=$(aws cloudformation describe-stacks \
    --stack-name $node_group_resources_name \
    --query='Stacks[].Outputs[?OutputKey==`NodeInstanceRole`].OutputValue'
 \
    --output text)
launch_template=$(aws cloudformation describe-stacks \
    --stack-name $node_group_resources_name \
    --query='Stacks[].Outputs[?OutputKey==`LaunchTemplateID`].OutputValue'
 \
    --output text)
```

   b. Créez un groupe de nœuds géré qui utilise le modèle de lancement et le rôle IAM de nœud créés à l'étape précédente.

```
aws eks create-nodegroup \
    --cluster-name $cluster_name \
    --nodegroup-name $node_group_name \
    --node-role $node_instance_role \
    --subnets $subnet_id \
```

```
    --launch-template id=$launch_template,version=1
```

c. Vérifiez que les nœuds ont été créés.

```
aws eks describe-nodegroup \
   --cluster-name ${cluster_name} \
   --nodegroup-name ${node_group_name} | jq -r .nodegroup.status
```

Ne continuez pas tant que l'état renvoyé par la commande précédente présente `ACTIVE`. Plusieurs minutes peuvent s'écouler avant que les nœuds ne soient prêts.

7. Si vous avez choisi un type d'instance GPU, vous devez déployer le plug-in de périphérique NVIDIA pour Kubernetes. Remplacez *vX.X.X* par la version NVIDIA/k8s-device-plugin souhaitée avant d'exécuter la commande suivante.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
```

4. Déployez le plugin de l'appareil EFA Kubernetes.

Le plugin de l'appareil EFA Kubernetes détecte et annonce les interfaces EFA comme ressources allouables à Kubernetes. Une application peut utiliser le type `vpc.amazonaws.com/efa` de ressource étendue dans une spécification de requête de Pod comme le processeur et la mémoire. Pour plus d'informations, consultez Consommation des ressources étendues dans la documentation Kubernetes. Une fois demandé, le plugin attribue et monte automatiquement une interface EFA au Pod. L'utilisation du plugin de l'appareil simplifie la configuration EFA et ne nécessite pas de Pod pour s'exécuter en mode privilégié.

```
helm repo add eks https://aws.github.io/eks-chart
helm install aws-efa-k8s-device-plugin --namespace kube-system eks/aws-efa-k8s-device-plugin
```

(Facultatif) Déployez un exemple d'application compatible EFA

Déployer l'opérateur MPI Kubeflow

Pour les tests NCCL, vous pouvez appliquer l'opérateur MPI Kubeflow. L'opérateur MPI facilite l'exécution d'un entraînement distribué de style AllReduce sur Kubernetes. Pour plus d'informations, consultez Opérateur MPI sur GitHub.

```
kubectl apply -f https://raw.githubusercontent.com/kubeflow/mpi-operator/master/deploy/v2beta1/mpi-operator.yaml
```

Exécuter le test de performance NCCL multi-nœuds pour vérifier GPUDirectrDMA/EFA

Pour vérifier les performances de NCCL avec GPUDirectRDMA sur EFA, exécutez le test de performance NCCL standard. Pour plus d'informations, consultez le référentiel officiel des tests NCCL sur GitHub. Vous pouvez utiliser l'exemple de ficher Dockerfile inclus avec ce test, déjà créé pour à la fois NVIDIA CUDA 11.2 et la dernière version d'EFA.

Vous pouvez également télécharger une AWS Docker image disponible depuis un dépôt Amazon ECR.

> **Important**
>
> Une considération importante requise pour adopter EFA avec Kubernetes est la configuration et la gestion de Huge Pages en tant que ressource dans le cluster. Pour plus d'informations, consultez Gérer Huge Pages dans la documentation Kubernetes. Les instances Amazon EC2 avec le pilote EFA installé pré-allouent 5128 2M Huge Pages, que vous pouvez demander en tant que ressources à utiliser dans les spécifications de votre tâche.

Procédez comme indiqué ci-dessous pour exécuter un test de performance NCCL à deux nœuds. Dans l'exemple de tâche de tests NCCL, chaque master demande huit GPU, 5210 Mi de hugepages-2Mi, quatre EFA et 8000 Mi de mémoire, ce qui signifie que chaque master consomme toutes les ressources d'une instance `p4d.24xlarge`.

1. Créez la tâche des tests NCCL.

   ```
   kubectl apply -f https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/examples/simple/nccl-efa-tests.yaml
   ```

   L'exemple qui suit illustre un résultat.

   mpijob.kubeflow.org/ a été créé nccl-tests-efa
2. Affichez votre Pods en cours d'exécution.

   ```
   kubectl get pods
   ```

L'exemple qui suit illustre un résultat.

```
NAME                              READY   STATUS     RESTARTS   AGE
nccl-tests-efa-launcher-nbql9  0/1     Init:0/1   0          2m49s
nccl-tests-efa-worker-0           1/1     Running    0          2m49s
nccl-tests-efa-worker-1           1/1     Running    0          2m49s
```

L'opérateur MPI crée un Pod de lancement et 2 Pods de travail (un sur chaque nœud).

3. Affichez le journal du Pod `efa-launcher`. Remplacez *wzr8j* par la valeur de votre sortie.

```
kubectl logs -f nccl-tests-efa-launcher-nbql9
```

Pour obtenir plus d'exemples, consultez le référentiel des exemples d'EFA Amazon EKS sur GitHub.

Inférence de machine learning à l'aide de AWS Inferentia

Cette rubrique décrit comment créer un cluster Amazon EKS avec des nœuds de travail exécutant des instances Amazon EC2 Inf1 et (éventuellement) déployer un exemple d'application. Les instances Amazon EC2 Inf1 sont alimentées par des puces AWS Inferentia, conçues sur mesure AWS pour fournir des performances élevées et une inférence à moindre coût dans le cloud. Les modèles d'apprentissage automatique sont déployés sur des conteneurs à l'aide de AWS Neuron, un kit de développement logiciel (SDK) spécialisé composé d'un compilateur, d'un environnement d'exécution et d'outils de profilage qui optimisent les performances d'inférence par apprentissage automatique des puces Inferentia. AWS Neuron prend en charge les frameworks d'apprentissage automatique populaires tels que TensorFlow, PyTorch, et MXnet.

> **Note**
>
> Les ID logiques de l'appareil Neuron doivent être contigus. Si un Pod demandant plusieurs appareils Neuron est programmé sur un `inf1.6xlarge` ou un type d'instance `inf1.24xlarge` (qui a plusieurs appareils Neuron), ce Pod ne démarrera pas si le planificateur Kubernetes sélectionne des ID d'appareil non contigus. Pour de plus amples informations, veuillez consulter Les ID logiques d'appareils doivent être contigus sur GitHub.

Prérequis

- Installez `eksctl` sur votre ordinateur. Si vous ne l'avez pas installé, consultez Installation dans la documentation `eksctl`.
- Installez `kubectl` sur votre ordinateur. Pour plus d'informations, consultez Installation ou mise à jour de `kubectl`.
- (Facultatif) Installez `python3` sur votre ordinateur. S'il n'est pas installé, consultez les téléchargements Python pour obtenir des instructions d'installation.

Créer un cluster

Pour créer un cluster avec des nœuds d'instances Amazon EC2 Inf1

1. Créez un cluster avec des nœuds d'instances Amazon EC2 Inf1. Vous pouvez remplacer *`inf1.2xlarge`* par n'importe quel type d'instance Inf1. L'utilitaire `eksctl` détecte que vous lancez un groupe de nœuds avec un type d'instance `Inf1` et démarre vos nœuds à l'aide de l'une des AMI Amazon Linux optimisées et accélérées.

   > **Note**
   >
   > Vous ne pouvez pas utiliser de rôles IAM pour les comptes de service avec TensorFlow Serving.

   ```
   eksctl create cluster \
       --name inferentia \
       --region region-code \
       --nodegroup-name ng-inf1 \
       --node-type inf1.2xlarge \
       --nodes 2 \
       --nodes-min 1 \
       --nodes-max 4 \
       --ssh-access \
       --ssh-public-key your-key \
       --with-oidc
   ```

> **Note**
>
> Notez la valeur de la ligne suivante de la sortie. Elle est utilisée lors d'une étape ultérieure (facultative).
>
> ```
> [9]  adding identity "arn:aws:iam::111122223333:role/
> eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09" to auth
>  ConfigMap
> ```

Lorsque vous lancez un groupe de nœuds avec des `Inf1` instances, le plug-in du Kubernetes périphérique AWS Neuron est `eksctl` automatiquement installé. Ce plugin annonce les appareils Neuron comme une ressource système au planificateur Kubernetes, qui peut être demandée par un conteneur. Outre les stratégies IAM de nœud Amazon EKS par défaut, la stratégie d'accès en lecture seule Amazon S3 est ajoutée afin que l'exemple d'application, traité dans une étape ultérieure, puisse charger un modèle formé à partir d'Amazon S3.

2. Assurez-vous que tous les Pods ont démarré correctement.

```
kubectl get pods -n kube-system
```

Sortie abrégée :

```
NAME                                  READY   STATUS    RESTARTS   AGE
[...]
neuron-device-plugin-daemonset-6djhp   1/1     Running   0          5m
neuron-device-plugin-daemonset-hwjsj   1/1     Running   0          5m
```

(Facultatif) Déployez une image d'application TensorFlow Serving

Un modèle formé doit être compilé sur une cible Inferentia avant de pouvoir être déployé sur des instances Inferentia. Pour continuer, vous aurez besoin d'un TensorFlow modèle optimisé pour Neuron enregistré dans Amazon S3. Si vous n'en avez pas encore SavedModel, veuillez suivre le didacticiel pour créer un modèle ResNet 50 compatible avec Neuron et télécharger le résultat SavedModel sur S3. ResNet-50 est un modèle d'apprentissage automatique populaire utilisé pour les tâches de reconnaissance d'images. Pour plus d'informations sur la compilation de modèles Neuron,

consultez la section La puce AWS Inferentia avec DLAMI dans le guide du développeur. AWS Deep Learning AMI

L'exemple de manifeste de déploiement gère un conteneur de service d'inférence prédéfini TensorFlow fourni par AWS Deep Learning Containers. À l'intérieur du conteneur se trouvent le AWS Neuron Runtime et l'application TensorFlow Serving. Une liste complète des conteneurs Deep Learning préconstruits optimisés pour Neuron est conservée sur GitHub sous Images disponibles. Au démarrage, le DLC récupérera votre modèle sur Amazon S3, lancera Neuron TensorFlow Serving avec le modèle enregistré et attendra les demandes de prédiction.

Le nombre d'appareils Neuron alloués à votre application de service peut être ajusté en changeant la ressource `aws.amazon.com/neuron` dans le yaml de déploiement. Veuillez noter que la communication entre TensorFlow Serving et le runtime Neuron se fait via GRPC, ce qui nécessite de transmettre la `IPC_LOCK` capacité au conteneur.

1. Ajoutez la stratégie IAM `AmazonS3ReadOnlyAccess` au rôle d'instance de nœud créé à l'étape 1 de Créer un cluster . Ceci est nécessaire pour que l'application exemple puisse charger un modèle formé à partir de Amazon S3.

   ```
   aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
       --role-name eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09
   ```

2. Créez un fichier nommé `rn50_deployment.yaml` avec les contenus suivants. Mettez à jour le code régional et le chemin du modèle pour correspondre aux paramètres souhaités. Le nom du modèle est utilisé à des fins d'identification lorsqu'un client fait une demande au TensorFlow serveur. Cet exemple utilise un nom de modèle correspondant à un exemple de ResNet 50 scripts client qui sera utilisé ultérieurement pour envoyer des demandes de prédiction.

   ```
   aws ecr list-images --repository-name neuron-rtd --registry-id 790709498068 --
   region us-west-2
   ```

   ```
   kind: Deployment
   apiVersion: apps/v1
   metadata:
     name: eks-neuron-test
     labels:
       app: eks-neuron-test
       role: master
   spec:
   ```

```
  replicas: 2
  selector:
    matchLabels:
      app: eks-neuron-test
      role: master
  template:
    metadata:
      labels:
        app: eks-neuron-test
        role: master
    spec:
      containers:
        - name: eks-neuron-test
          image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-
neuron:1.15.4-neuron-py37-ubuntu18.04
          command:
            - /usr/local/bin/entrypoint.sh
          args:
            - --port=8500
            - --rest_api_port=9000
            - --model_name=resnet50_neuron
            - --model_base_path=s3://your-bucket-of-models/resnet50_neuron/
          ports:
            - containerPort: 8500
            - containerPort: 9000
          imagePullPolicy: IfNotPresent
          env:
            - name: AWS_REGION
              value: "us-east-1"
            - name: S3_USE_HTTPS
              value: "1"
            - name: S3_VERIFY_SSL
              value: "0"
            - name: S3_ENDPOINT
              value: s3.us-east-1.amazonaws.com
            - name: AWS_LOG_LEVEL
              value: "3"
          resources:
            limits:
              cpu: 4
              memory: 4Gi
              aws.amazon.com/neuron: 1
            requests:
              cpu: "1"
```

```
              memory: 1Gi
          securityContext:
            capabilities:
              add:
                - IPC_LOCK
```

3. Déployez le modèle.

```
kubectl apply -f rn50_deployment.yaml
```

4. Créez un fichier nommé `rn50_service.yaml` avec les contenus suivants. Les ports HTTP et gRPC sont ouverts pour accepter les demandes de prédiction.

```
kind: Service
apiVersion: v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
spec:
  type: ClusterIP
  ports:
    - name: http-tf-serving
      port: 8500
      targetPort: 8500
    - name: grpc-tf-serving
      port: 9000
      targetPort: 9000
  selector:
    app: eks-neuron-test
    role: master
```

5. Créez un Kubernetes service pour votre application de service TensorFlow modèle.

```
kubectl apply -f rn50_service.yaml
```

(Facultatif) Faites des prédictions par rapport à votre TensorFlow service de service

1. Pour tester localement, transférez le port gRPC au service `eks-neuron-test`.

```
kubectl port-forward service/eks-neuron-test 8500:8500 &
```

2. Créez un script Python appelé `tensorflow-model-server-infer.py` avec le contenu suivant. Ce script exécute l'inférence via gRPC, qui est une infrastructure de service.

```
import numpy as np
   import grpc
   import tensorflow as tf
   from tensorflow.keras.preprocessing import image
   from tensorflow.keras.applications.resnet50 import preprocess_input
   from tensorflow_serving.apis import predict_pb2
   from tensorflow_serving.apis import prediction_service_pb2_grpc
   from tensorflow.keras.applications.resnet50 import decode_predictions

   if __name__ == '__main__':
       channel = grpc.insecure_channel('localhost:8500')
       stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
       img_file = tf.keras.utils.get_file(
           "./kitten_small.jpg",
           "https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/
docs/images/kitten_small.jpg")
       img = image.load_img(img_file, target_size=(224, 224))
       img_array = preprocess_input(image.img_to_array(img)[None, ...])
       request = predict_pb2.PredictRequest()
       request.model_spec.name = 'resnet50_inf1'
       request.inputs['input'].CopyFrom(
           tf.make_tensor_proto(img_array, shape=img_array.shape))
       result = stub.Predict(request)
       prediction = tf.make_ndarray(result.outputs['output'])
       print(decode_predictions(prediction))
```

3. Exécutez le script pour soumettre des prédictions à votre service.

```
python3 tensorflow-model-server-infer.py
```

L'exemple qui suit illustre un résultat.

```
[[(u'n02123045', u'tabby', 0.68817204), (u'n02127052', u'lynx', 0.12701613),
 (u'n02123159', u'tiger_cat', 0.08736559), (u'n02124075', u'Egyptian_cat',
 0.063844085), (u'n02128757', u'snow_leopard', 0.009240591)]]
```

Gestion du cluster

Ce chapitre porte sur les rubriques suivantes relatives à la gestion de votre cluster. Vous pouvez également afficher des informations sur vos ressources Kubernetes avec la AWS Management Console.

- Le tableau de bord Kubernetes est une interface utilisateur Web à usage général pour les clusters Kubernetes. Il permet aux utilisateurs de gérer les applications exécutées dans le cluster et de les dépanner, ainsi que de gérer le cluster lui-même. Pour plus d'informations, consultez le référentiel GitHub du tableau de bord Kubernetes.
- Installation du serveur de métriques Kubernetes : le serveur de métriques Kubernetes est un agrégateur de données sur l'utilisation des ressources dans votre cluster. Il n'est pas déployé par défaut dans votre cluster, mais il est utilisé par les modules complémentaires de Kubernetes, tels que Kubernetes Dashboard et Horizontal Pod Autoscaler. Dans cette rubrique, vous allez apprendre à installer le serveur de métriques.
- Utilisation de Helm avec Amazon EKS : le gestionnaire de package Helm pour Kubernetes vous permet d'installer et de gérer des applications dans votre cluster Kubernetes. Cette rubrique vous aide à installer et à exécuter les binaires Helm, afin de pouvoir installer et gérer des graphiques à l'aide de la CLI Helm sur votre ordinateur local.
- Étiquetage de vos ressources Amazon EKS : pour vous aider à gérer vos ressources Amazon EKS, vous pouvez attribuer vos propres métadonnées à chaque ressource sous forme d'identifications. Cette rubrique décrit les identifications et explique comment les créer.
- Service quotas Amazon EKS : votre compte AWS dispose de quotas par défaut, anciennement appelés limites, pour chaque service AWS. Découvrez les quotas pour Amazon EKS et la manière de les augmenter.

Suivi des coûts

La surveillance des coûts est un aspect essentiel de la gestion de vos Kubernetes clusters sur Amazon EKS. En obtenant une meilleure visibilité sur les coûts de votre cluster, vous pouvez optimiser l'utilisation des ressources, définir des budgets et prendre des décisions basées sur les données concernant vos déploiements. Amazon EKS propose deux solutions de surveillance des coûts, chacune présentant ses propres avantages uniques, pour vous aider à suivre et à répartir vos coûts de manière efficace :

AWS Données de répartition des coûts de facturation pour Amazon EKS : cette fonctionnalité native s'intègre parfaitement à la console de AWS facturation, vous permettant d'analyser et de répartir les coûts à l'aide de la même interface et des mêmes flux de travail que ceux que vous utilisez pour les autres AWS services. Grâce à la répartition des coûts, vous pouvez obtenir des informations sur vos Kubernetes coûts directement en parallèle avec vos autres AWS dépenses, ce qui facilite l'optimisation globale des coûts dans l'ensemble de votre AWS environnement. Vous pouvez également tirer parti des fonctionnalités AWS de facturation existantes, telles que Cost Categories et Cost Anomaly Detection, pour améliorer encore vos capacités de gestion des coûts. Pour plus d'informations, consultez la section Comprendre les données de répartition des coûts partagés dans le Guide AWS de l'utilisateur de facturation.

Kubecost— Amazon EKS prend en charge Kubecost, un outil de surveillance des coûts Kubernetes. Kubecost propose une approche riche en fonctionnalités et native de Kubernetes en matière de surveillance des coûts, fournissant des ventilations détaillées des coûts par ressources Kubernetes, des recommandations d'optimisation des coûts, ainsi que des tableaux de bord et des rapports. out-of-the-box Kubecost récupère également des données de tarification précises en les intégrant au rapport sur les AWS coûts et l'utilisation, ce qui vous permet d'obtenir une vue précise de vos coûts Amazon EKS. Découvrez comment installer Kubecost.

AWS Facturation — Répartition des coûts fractionnés

Surveillance des coûts à l'aide des données de répartition des coûts AWS fractionnés pour Amazon EKS

Vous pouvez utiliser AWS les données de répartition des coûts pour Amazon EKS afin d'obtenir une visibilité précise des coûts pour vos clusters Amazon EKS. Cela vous permet d'analyser, d'optimiser et de rétrofacturer les coûts et l'utilisation de vos Kubernetes applications. Vous répartissez les coûts des applications entre les différentes unités commerciales et les équipes en fonction des ressources de processeur et de mémoire Amazon EC2 consommées par votre Kubernetes application. Les données de répartition des coûts fractionnées pour Amazon EKS offrent une visibilité sur le coût par module et vous permettent d'agréger les données de coût par module à l'aide de l'espace de noms, du cluster et d'autres Kubernetes primitives. Vous trouverez ci-dessous des exemples de Kubernetes primitives que vous pouvez utiliser pour analyser les données de répartition des coûts Amazon EKS.

- Nom du cluster
- Déploiement
- Espace de noms
- Nœud

- Nom de la charge de travail
- Type de charge de travail

Pour plus d'informations sur l'utilisation des données de répartition des coûts partagés, voir Comprendre les données de répartition des coûts partagés dans le Guide AWS de l'utilisateur de facturation.

Configuration des rapports sur les coûts et l'utilisation

Vous pouvez activer les données de répartition des coûts pour EKS dans la console de gestion des coûts ou dans les AWS SDK. AWS Command Line Interface

Utilisez ce qui suit pour les données de répartition des coûts fractionnés :

1. Choisissez de fractionner les données de répartition des coûts. Pour plus d'informations, consultez la section Activation des données de répartition des coûts fractionnés dans le Guide de AWS Cost and Usage Report l'utilisateur.
2. Incluez les données dans un rapport nouveau ou existant.
3. Consultez le rapport. Vous pouvez utiliser la console de Facturation et gestion des coûts ou consulter les fichiers de rapports dans Amazon Simple Storage Service.

Kubecost

Amazon EKS prend en charge Kubecost, que vous pouvez utiliser pour suivre vos coûts ventilés par ressources Kubernetes, y compris les Pods, les nœuds, les espaces de noms et les étiquettes. En tant qu'administrateur et responsable financier de la plateforme Kubernetes, vous pouvez utiliser Kubecost pour visualiser la répartition des frais Amazon EKS, répartir les coûts et facturer les unités organisationnelles telles que les équipes d'application. Vous pouvez fournir à vos équipes internes et à vos unités commerciales des données de coûts transparentes et précises basées sur leur AWS facture réelle. En outre, vous pouvez également obtenir des recommandations personnalisées pour l'optimisation des coûts en fonction de leur environnement d'infrastructure et des modèles d'utilisation au sein de leurs clusters. Pour plus d'informations sur Kubecost, consultez la documentation Kubecost.

Amazon EKS fournit une offre groupée AWS optimisée Kubecost pour la visibilité des coûts du cluster. Vous pouvez utiliser vos contrats de AWS support existants pour obtenir de l'aide.

Prérequis

- Un cluster Amazon EKS existant. Pour en déployer un, consultez Démarrer avec Amazon EKS. Le cluster doit disposer de nœuds Amazon EC2, car vous ne pouvez pas exécuter Kubecost sur des nœuds Fargate.
- L'outil de ligne de commande `kubectl` est installé sur votre appareil ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Helm version 3.9.0 ou ultérieure configurée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour Helm, consultez the section called "Utilisation de Helm".
- Si votre cluster est une version `1.23` ou ultérieure, vous devez avoir le the section called "Pilote CSI Amazon EBS" installé sur votre cluster.

Pour installer Kubecost

1. Déterminez la version de Kubecost à installer. Vous pouvez consulter les versions disponibles sur kubecost/cost-analyzer dans la galerie publique Amazon ECR. Pour plus d'informations sur la compatibilité des Kubecost versions et Amazon EKS, consultez les exigences environnementales dans la documentation de Kubecost.
2. Installez Kubecost à l'aide de la commande suivante. *Remplacez kubecost-version par la valeur extraite de l'ECR, telle que 1.108.1.*

   ```
   helm upgrade -i kubecost oci://public.ecr.aws/kubecost/cost-analyzer --
   version kubecost-version \
       --namespace kubecost --create-namespace \
       -f https://raw.githubusercontent.com/kubecost/cost-analyzer-helm-chart/develop/
   cost-analyzer/values-eks-cost-monitoring.yaml
   ```

   Kubecost publie régulièrement de nouvelles versions. Vous pouvez mettre à jour votre version à l'aide de la mise à niveau `helm upgrade`. Par défaut, l'installation inclut un serveur local Prometheus et `kube-state-metrics`. Vous pouvez personnaliser votre déploiement pour utiliser Amazon Managed Service for Prometheus en suivant la documentation disponible dans Integrating with Amazon EKS cost monitoring. Pour obtenir la liste de tous les autres paramètres que vous pouvez configurer, consultez l'exemple de fichier de configuration sur GitHub.

3. Assurez-vous que les Pods requis fonctionnent.

```
kubectl get pods -n kubecost
```

L'exemple qui suit illustre un résultat.

```
NAME                                          READY   STATUS    RESTARTS   AGE
kubecost-cost-analyzer-b9788c99f-5vj5b        2/2     Running   0          3h27m
kubecost-kube-state-metrics-99bb8c55b-bn2br   1/1     Running   0          3h27m
kubecost-prometheus-server-7d9967bfc8-9c8p7   2/2     Running   0          3h27m
```

4. Sur votre appareil, activez la redirection de port pour afficher le tableau de bord Kubecost.

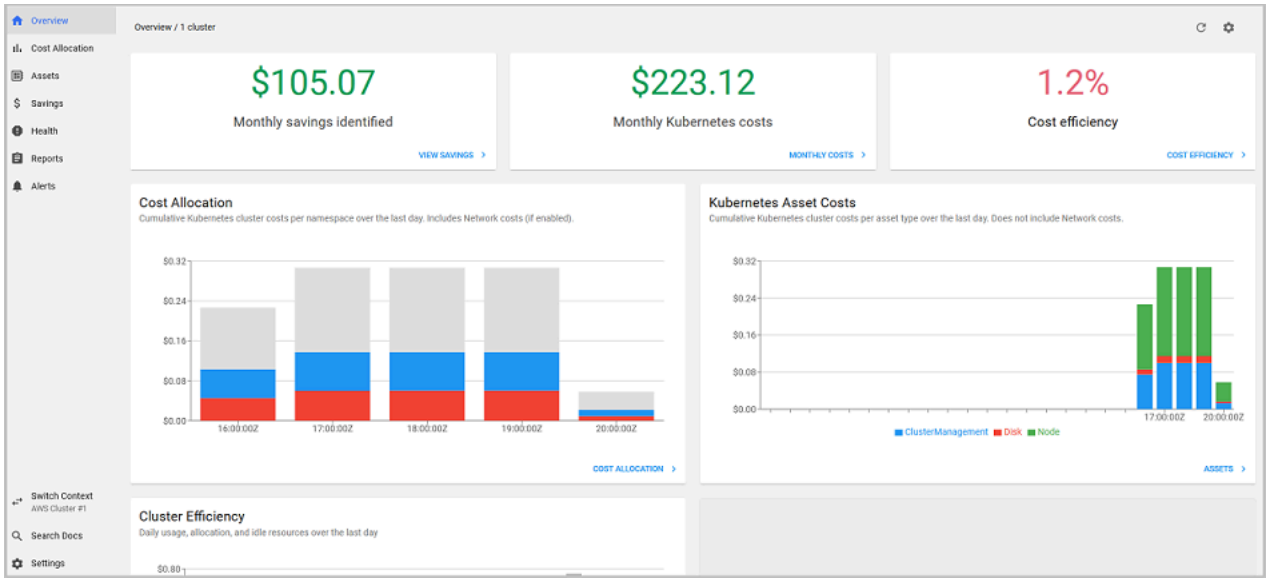
```
kubectl port-forward --namespace kubecost deployment/kubecost-cost-analyzer 9090
```

Sinon, vous pouvez utiliser AWS Load Balancer Controller pour exposer Kubecost et utiliser Amazon Cognito pour les authentifications, les autorisations et la gestion des utilisateurs. Pour plus d'informations, consultez Comment utiliser Application Load Balancer et comment authentifier Amazon Cognito les utilisateurs pour vos Kubernetes applications Web.

5. Depuis le même appareil qui vous a servi à effectuer l'étape précédente, ouvrez un navigateur Web et entrez l'adresse suivante.

```
http://localhost:9090
```

Vous verrez la page de présentation de Kubecost dans votre navigateur. La collecte des métriques peut prendre 5 à 10 minutes pour Kubecost. Vous pouvez voir vos dépenses Amazon EKS, y compris les coûts cumulés des clusters, les coûts des actifs Kubernetes associés et les dépenses agrégées mensuelles.



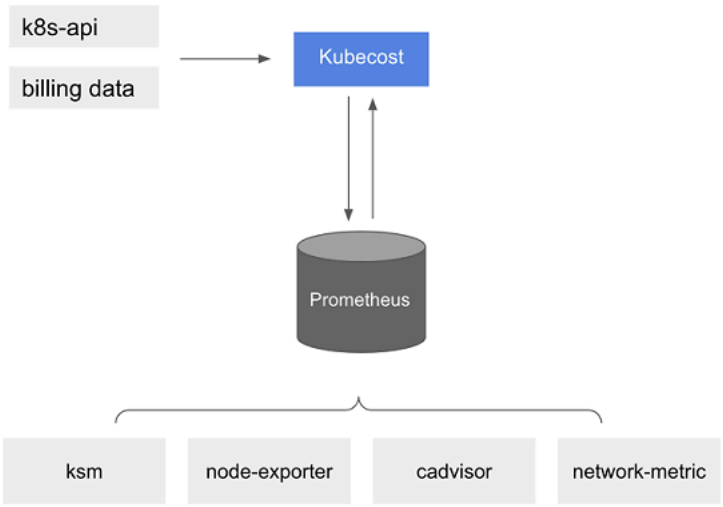
6. Pour suivre les coûts au niveau du cluster, balisez vos ressources Amazon EKS pour la facturation. Pour plus d'informations, consultez Identification de vos ressources pour facturation.

Vous pouvez également afficher les informations suivantes en les sélectionnant dans le volet gauche du tableau de bord :

- Allocation des coûts : Consultez les coûts mensuels d'Amazon EKS et les coûts cumulés pour chacun de vos espaces de noms et d'autres dimensions au cours des sept derniers jours. Cela est utile pour comprendre quels secteurs de votre application contribuent aux dépenses d'Amazon EKS.
- Ressources : Consultez les coûts des ressources d'infrastructure AWS qui sont associés à vos ressources Amazon EKS.

Fonctionnalités supplémentaires

- Métriques des coûts d'export : Le suivi optimisé des coûts d'Amazon EKS est déployé avec Kubecost et Prometheus, constituant un système de suivi open source et une base de données de séries temporelles. Kubecost lit la métrique depuis Prometheus puis effectue des calculs de répartition des coûts et réécrit les indicateurs dans Prometheus. Le front-end de Kubecost lit les métriques depuis Prometheus et les affiche sur l'interface utilisateur de Kubecost. L'architecture est illustrée dans le schéma suivant.



Avec Prometheus préinstallé, vous pouvez écrire des requêtes pour intégrer des données Kubecost à votre système de business intelligence actuel pour une analyse plus approfondie. Vous pouvez également l'utiliser comme source de données pour votre tableau de bord Grafana pour afficher les coûts des clusters Amazon EKS que vos équipes internes connaissent bien. Pour en savoir plus sur la façon d'écrire Prometheus des requêtes, consultez le `readme` fichier de Prometheusconfiguration GitHub ou utilisez les exemples de modèles Grafana JSON du référentiel Kubecost Github comme références.

- AWS Cost and Usage Report intégration — Pour effectuer des calculs de répartition des coûts pour votre cluster Amazon EKS, Kubecost extrait les informations de tarification publiques Services AWS et les AWS ressources de l'API AWS Price List. Vous pouvez également intégrer Kubecost AWS Cost and Usage Reportpour améliorer la précision des informations tarifaires spécifiques à votre Compte AWS. Ces informations incluent les programmes de remise pour les entreprises, l'utilisation d'instances réservées, les plans d'épargne et l'utilisation ponctuelle. Pour en savoir plus sur le fonctionnement de l' AWS Cost and Usage Report intégration, consultez la section Intégration de la facturation dans le AWS cloud dans la Kubecost documentation.

Supprimez Kubecost

Vous pouvez supprimer Kubecost de votre cluster à l'aide des commandes suivantes.

```
helm uninstall kubecost --namespace kubecost
kubectl delete ns kubecost
```

Questions fréquentes (FAQ)

Consultez les questions et réponses courantes suivantes concernant l'utilisation de Kubecost avec Amazon EKS.

Quelle est la différence entre l'offre groupée personnalisée Kubecost et la version gratuite de Kubecost (également appelée OpenCost) ?

AWS et Kubecost ont collaboré pour proposer une version personnalisée deKubecost. Cette version inclut un sous-ensemble de fonctions commerciales sans frais supplémentaires. Consultez le tableau suivant pour connaître les fonctionnalités incluses dans l'offre groupée personnalisée de Kubecost.

Fonctionnalité	Offre gratuite Kubecost	Offre groupée personnalisée Kubecost optimisée pour Amazon EKS	Kubecost Enterprise
Déploiement	Hébergé par l'utilisateur	Hébergé par l'utilisateur	Hébergé par l'utilisateur ou hébergé par Kubecost (SaaS)
Nombre de clusters pris en charge	Illimité	Illimité	Illimité
Bases de données prises en charge	Prometheus local	Prometheus local ou Amazon Managed Service for Prometheus	Prometheus, Amazon Managed Service for Prometheus, Cortex ou Thanos
Prise en charge de la conservation des bases de données	15 jours	Données historiques illimitées	Données historiques illimitées
Conservation d'API Kubecost (ETL)	15 jours	15 jours	Données historiques illimitées
Visibilité des coûts du cluster	Clusters uniques	Multicluster unifié	Multicluster unifié

Fonctionnalité	Offre gratuite Kubecost	Offre groupée personnalisée Kubecost optimisée pour Amazon EKS	Kubecost Enterprise
Visibilité du cloud hybride	-	Clusters Amazon EKS et Amazon EKS Anywhere	Support multicloud et cloud hybride
Alertes et rapports récurrents	-	Prise en charge des alertes d'efficacité, des alertes de budget, des alertes de modification des dépenses, et bien d'autres encore	Prise en charge des alertes d'efficacité, des alertes de budget, des alertes de modification des dépenses, et bien d'autres encore
Rapports enregistrés	-	Rapports utilisant des données sur 15 jours	Rapports utilisant des données historiques illimitées
Intégration de la facturation dans le cloud	Nécessaire pour chaque cluster individuel	Assistance tarifaire personnalisée pour AWS (y compris plusieurs clusters et plusieurs comptes)	Assistance tarifaire personnalisée pour AWS (y compris plusieurs clusters et plusieurs comptes)
Recommandations en matière d'épargne	Informations sur un seul cluster	Informations sur un seul cluster	Informations sur plusieurs clusters
Gouvernance : audits	-	-	Audit des événements liés aux coûts historiques
Prise en charge de l'authentification unique (SSO)	-	Prise en charge d'Amazon Cognito	Okta, Auth0, PingID, KeyCloak

Fonctionnalité	Offre gratuite Kubecost	Offre groupée personnalisée Kubecost optimisée pour Amazon EKS	Kubecost Enterprise
Contrôle d'accès basé sur les rôles (RBAC) avec SAML **2.0**	-	-	Okta, Auth0, PingID, Keycloak
Formation et intégration en entreprise	-	-	Formation et intégration FinOps complètes

Qu'est-ce que la fonctionnalité de conservation d'API Kubecost (ETL) ?

La fonctionnalité ETL Kubecost agrège et organise les métriques pour rendre visible les coûts à différents niveaux de granularité (comme `namespace-level`, `pod-level` et `deployment-level`). Avec l'offre groupée Kubecost personnalisée, les clients obtiennent des données et des informations issues des métriques des 15 derniers jours.

Qu'est-ce que la fonctionnalité d'alertes et de rapports récurrents ? Quels sont les alertes et les rapports inclus ?

Les alertes Kubecost permettent aux équipes de recevoir des mises à jour sur les dépenses Kubernetes en temps réel ainsi que sur les dépenses liées au cloud. Les rapports récurrents permettent aux équipes de recevoir des vues personnalisées des dépenses Kubernetes historiques et liées au cloud. Les deux peuvent être configurés à l'aide de l'interface utilisateur de Kubecost ou des valeurs Helm. Ils prennent en charge l'e-mail, Slack et Microsoft Teams.

Que contiennent les rapports enregistrés ?

Les rapports enregistrés Kubecost sont des vues prédéfinies des métriques de coûts et d'efficacité. Ils incluent le coût par cluster, l'espace de noms, l'étiquette et bien plus encore.

Qu'est-ce que l'intégration de la facturation dans le cloud ?

L'intégration aux API AWS de facturation Kubecost permet d'afficher out-of-cluster les coûts (comme Amazon S3). De plus, cela permet à Kubecost de rapprocher les prévisions internes de

Kubecost dans le cluster avec les données de facturation réelles pour tenir compte de l'utilisation des instances Spot, des plans d'épargne Savings Plans et des remises d'entreprise.

Que comprennent les recommandations en matière d'épargne ?

Kubecost offre des informations et une automatisation pour aider les utilisateurs à optimiser leur infrastructure Kubernetes et leurs dépenses.

Cette fonctionnalité entraîne-t-elle des frais ?

Non. Vous pouvez utiliser cette version de Kubecost sans frais supplémentaires. Si vous souhaitez des Kubecost fonctionnalités supplémentaires qui ne sont pas incluses dans cette offre groupée, vous pouvez acheter une licence d'entreprise Kubecost par le AWS Marketplace biais ou Kubecost directement auprès de.

Le support est-il disponible ?

Oui. Vous pouvez ouvrir un dossier d'assistance auprès de l' AWS Support équipe de Contact AWS.

Ai-je besoin d'une licence pour utiliser les fonctions Kubecost fournies par l'intégration Amazon EKS ?

Non.

Puis-je l'intégrer AWS Cost and Usage Report pour obtenir Kubecost des rapports plus précis ?

Oui. Vous pouvez configurer Kubecost pour ingérer des données provenant de AWS Cost and Usage Report et obtenir une visibilité précise des coûts, y compris les remises, la tarification Spot, les prix des instances réservées, etc. Pour plus d'informations, consultez la section Intégration de la facturation dans le AWS cloud dans la Kubecost documentation.

Cette version prend-elle en charge la gestion des coûts des clusters Kubernetes autogérés sur Amazon EC2 ?

Non. Cette version est uniquement compatible avec les clusters Amazon EKS.

Kubecost peut-il suivre les coûts Amazon EKS sur AWS Fargate ?

Kubecost fait tout son possible pour afficher la visibilité des coûts de cluster pour Amazon EKS sur Fargate, mais la précision est moins élevée qu'avec Amazon EKS sur Amazon EC2. Cela est principalement dû à la différence de facturation de l'utilisation. Avec Amazon EKS sur Fargate, les ressources consommées vous sont facturées. Avec Amazon EKS sur les nœuds Amazon EC2, les ressources allouées vous sont facturées. Kubecost calcule le coût d'un nœud Amazon EC2 en

fonction de la spécification du nœud, qui inclut le processeur, la RAM et le magasin éphémère. Avec Fargate, les coûts sont calculés en fonction des ressources demandées pour les Pods Fargate.

Comment puis-je obtenir les mises à jour et les nouvelles versions de Kubecost ?

Vous pouvez mettre à jour votre version de Kubecost en utilisant les procédures de mise à niveau standard de Helm. Les dernières versions se trouvent dans la Galerie publique Amazon ECR.

La CLI **`kubectl-cost`** est-elle prise en charge ? Comment l'installer ?

Oui. `Kubectl-cost` est un outil open source par Kubecost (licence Apache 2.0) qui fournit un accès par la CLI aux métriques de répartition des coûts Kubernetes. Pour procéder à l'installation`kubectl-cost`, reportez-vous à la section Installation sur GitHub.

L'interface utilisateur de Kubecost est-elle prise en charge ? Comment y accéder ?

Kubecost fournit un tableau de bord web accessible via le réacheminement de port `kubectl`, une entrée ou un équilibreur de charge. Vous pouvez également utiliser le AWS Load Balancer Controller pour exposer Kubecost et utiliser Amazon Cognito pour les authentifications, les autorisations et la gestion des utilisateurs. Pour plus d'informations, consultez Comment utiliser Application Load Balancer et Amazon Cognito pour authentifier les utilisateurs de Kubernetes vos applications Web sur le blog. AWS

Amazon EKS Anywhere est-il pris en charge ?

Non.

Installation du serveur de métriques Kubernetes

Le serveur de métriques Kubernetes est un agrégateur de données d'utilisation des ressources dans votre cluster. Il n'est pas déployé par défaut dans les clusters Amazon EKS. Pour de plus amples informations, veuillez consulter Configuration du serveur de métriques Kubernetes sur GitHub. Le serveur de métriques est couramment utilisé par les autres modules Kubernetes, tels que le Horizontal Pod Autoscaler ou le tableau de bord Kubernetes. Pour plus d'informations, consultez Pipeline de métriques de ressource dans la documentation Kubernetes. Cette rubrique explique comment déployer le serveur de métriques Kubernetes sur votre cluster Amazon EKS.

> **Important**
>
> Les métriques sont destinées à point-in-time l'analyse et ne constituent pas une source précise pour l'analyse historique. Elles ne peuvent pas être utilisées comme solution de

surveillance ou à d'autres fins de mise à l'échelle non auto. Pour de plus amples informations sur la surveillance, consultez Observabilité dans Amazon EKS.

Déployer le serveur de métriques

1. Déployez le serveur de métriques à l'aide de la commande suivante :

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

   Si vous utilisez Fargate, vous devez modifier ce fichier. Dans la configuration par défaut, le serveur de métriques utilise le port 10250. Ce port est réservé sur Fargate. Remplacez les références au port 10250 dans components.yaml par un autre port, tel que 10251.

2. Vérifiez que le déploiement du `metrics-server` exécute le nombre souhaité de Pods avec la commande suivante.

```
kubectl get deployment metrics-server -n kube-system
```

   L'exemple qui suit illustre un résultat.

```
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
metrics-server   1/1     1            1           6m
```

Utilisation de Helm avec Amazon EKS

Le gestionnaire de package Helm pour Kubernetes vous permet d'installer et de gérer des applications dans votre cluster Kubernetes. Pour plus d'informations, consultez la documentation Helm. Cette rubrique vous aide à installer et à exécuter les fichiers binaires Helm, afin de pouvoir installer et gérer des charts à l'aide de la CLI Helm sur votre système local.

> **Important**
>
> Pour pouvoir installer des Charts Helm dans votre cluster, vous devez configurer `kubectl` pour qu'il fonctionne avec Amazon EKS. Si vous ne l'avez pas encore fait, veuillez consulter Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS avant de

poursuivre. Si la commande suivante aboutit pour votre cluster, votre configuration est correcte.

```
kubectl get svc
```

Pour installer les fichiers binaires Helm sur votre système local

1. Exécutez la commande adaptée à votre système d'exploitation client.

   - Si vous utilisez macOS avec Homebrew, installez les fichiers binaires à l'aide de la commande suivante.

     ```
     brew install helm
     ```

   - Si vous utilisez Windows avec Chocolatey, installez les fichiers binaires à l'aide de la commande suivante.

     ```
     choco install kubernetes-helm
     ```

   - Si vous utilisez Linux, installez les binaires avec les commandes suivantes.

     ```
     curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 >
      get_helm.sh
     chmod 700 get_helm.sh
     ./get_helm.sh
     ```

     Note

     Si vous recevez un message indiquant que `openssl` doit d'abord être installé, vous pouvez utiliser la commande suivante pour l'installer.

     ```
     sudo yum install openssl
     ```

2. Pour récupérer le nouveau fichier binaire dans votre `PATH`, fermez votre fenêtre de terminal actuelle et ouvrez une nouvelle fenêtre.
3. Vérifiez la version de Helm que vous avez installée.

```
helm version | cut -d + -f 1
```

L'exemple qui suit illustre un résultat.

```
v3.9.0
```

4. À ce stade, vous pouvez exécuter toutes les commandes Helm (telles que `helm install chart-name`) pour installer, modifier, supprimer ou interroger les charts Helm dans votre cluster. Si vous découvrez Helm et que vous n'avez pas de chart spécifique à installer, vous pouvez :

   - Tester en installant un exemple de chart. Reportez-vous à la rubrique sur l'installation d'un exemple de chart dans le Guide Quickstart Helm.
   - Créez un exemple de chart et envoyez-le à Amazon ECR. Pour plus d'informations, consultez la rubrique relative à l'envoi d'un chart Helm dans le Guide de l'utilisateur Amazon Elastic Container Registry.
   - Installez un graphique Amazon EKS depuis le GitHub dépôt eks-charts ou depuis. ArtifactHub

Étiquetage de vos ressources Amazon EKS

Vous pouvez utiliser les balises pour vous aider à gérer vos ressources Amazon EKS. Cette rubrique fournit une vue d'ensemble de la fonction des identifications et indique comment créer des identifications.

Rubriques

- Principes de base des étiquettes
- Balisage de vos ressources
- Restrictions liées aux étiquettes
- Identification de vos ressources pour facturation
- Gestion des étiquettes à l'aide de la console
- Gestion des identifications à l'aide de la CLI, de l'API ou de eksctl

> **Note**
>
> Les balises sont un type de métadonnées distinct des étiquettes et annotations Kubernetes. Pour plus d'informations sur ces autres types de métadonnées, consultez les sections suivantes dans la documentation Kubernetes :
>
> - Étiquettes et sélecteurs
> - Annotations

Principes de base des étiquettes

Une étiquette est une étiquette que vous attribuez à une AWS ressource. Chaque balise est constituée d'une clé et d'une valeur facultative.

Les tags vous permettent de classer vos AWS ressources par catégories. Par exemple, vous pouvez classer les ressources par objectif, propriétaire ou environnement. Lorsque vous avez de nombreuses ressources de même type, vous pouvez utiliser les balises que vous avez attribuées à une ressource spécifique pour identifier cette dernière rapidement. Par exemple, vous pouvez définir un ensemble d'identifications pour vos clusters Amazon EKS afin de vous aider à suivre le propriétaire et le niveau de pile de chaque cluster. Nous vous recommandons de concevoir un ensemble cohérent de clés de balise pour chaque type de ressource. Vous pouvez rechercher et filtrer les ressources en fonction des balises que vous ajoutez.

Une fois que vous avez ajouté une balise, vous pouvez modifier les clés et valeurs de balise ou supprimer les balises d'une ressource à tout moment. Si vous supprimez une ressource, ses balises sont également supprimées.

Les identifications n'ont pas de signification sémantique pour Amazon EKS et sont interprétées strictement comme des chaînes de caractères. Vous pouvez définir la valeur d'une balise à une chaîne vide. Toutefois, vous ne pouvez pas définir la valeur d'une balise sur zéro. Si vous ajoutez une balise ayant la même clé qu'une balise existante sur cette ressource, la nouvelle valeur remplace l'ancienne valeur.

Si vous utilisez AWS Identity and Access Management (IAM), vous pouvez contrôler quels utilisateurs de votre AWS compte sont autorisés à gérer les tags.

Balisage de vos ressources

Les ressources Amazon EKS suivantes prennent en charge les balises :

- clusters
- groupes de nœuds gérés
- profils Fargate

Vous pouvez baliser ces ressources à l'aide des éléments suivants :

- Si vous utilisez la console Amazon EKS, vous pouvez à tout moment appliquer des identifications aux ressources nouvelles ou existantes. Pour ce faire, utilisez la commande Identifications sur la page de ressources correspondante. Pour plus d'informations, consultez Gestion des étiquettes à l'aide de la console.
- Si vous utilisez `eksctl`, vous pouvez appliquer des balises aux ressources lorsqu'elles sont créées à l'aide de l'option `--tags`.
- Si vous utilisez l' AWS CLI API Amazon EKS ou un AWS SDK, vous pouvez appliquer des balises aux nouvelles ressources en utilisant le `tags` paramètre de l'action d'API correspondante. Vous pouvez également appliquer des identifications aux ressources à l'aide de l'action d'API `TagResource`. Pour plus d'informations, consultez TagResource.

Lorsque vous exécutez certaines actions de création des ressources, vous pouvez également spécifier des balises pour la ressource en même temps que vous la créez. Si les balises ne peuvent pas être appliquées pendant la création de la ressource, la ressource ne peut pas être créée. Ce mécanisme garantit que les ressources que vous prévoyez de baliser sont créées avec les balises que vous spécifiez ou ne sont pas créées du tout. Si vous balisez des ressources lorsque vous les créez, vous n'avez pas besoin d'exécuter de scripts de balisage personnalisés après avoir créé les ressources.

Les balises ne sont pas propagées vers les autres ressources associées à la ressource que vous créez. Par exemple, les balises de profil Fargate ne se propagent pas aux autres ressources qui sont associées au profil Fargate, telles que les Pods qui sont programmés avec ce dernier.

Restrictions liées aux étiquettes

Les restrictions suivantes s'appliquent aux balises :

- Un maximum de 50 balises peut être associé à une ressource.
- Les clés de balises ne peuvent pas être répétées pour une même ressource. Chaque clé de balise doit être unique et ne peut avoir qu'une seule valeur.
- Les clés peuvent contenir jusqu'à 128 caractères en UTF-8.
- Les valeurs peuvent contenir jusqu'à 256 caractères en UTF-8.
- Si plusieurs Services AWS ressources utilisent votre schéma de balisage, limitez les types de caractères que vous utilisez. Certains services peuvent présenter des restrictions sur les caractères autorisés. Les caractères généralement autorisés sont les lettres, les chiffres, les espaces et les caractères suivants : **+ - = . _ : / @**.
- Les clés et valeurs de balise sont sensibles à la casse.
- N'utilisez pas `aws:`, `AWS:` ou n'importe quelle combinaison de majuscules ou minuscules de ce préfixe pour des clés ou des valeurs Ils sont réservés uniquement à AWS l'usage. Vous ne pouvez pas modifier ni supprimer des clés ou valeurs d'étiquette ayant ce préfixe. Les balises comportant ce préfixe ne sont pas prises en compte dans votre tags-per-resource limite.

Identification de vos ressources pour facturation

Lorsque vous appliquez des balises à des clusters Amazon EKS, vous pouvez les utiliser pour la répartition des coûts dans vos Cost & Usage Reports (Rapports sur les coûts et l'utilisation). Les données de mesure de vos rapports sur les coût et l'utilisation illustrent l'utilisation parmi tous vos clusters Amazon EKS. Pour en savoir plus, consultez rapport sur les coûts et l'utilisation AWS dans le Guide de l'utilisateur des AWS Billing s.

La balise de répartition des coûts AWS générée, en particulier`aws:eks:cluster-name`, vous permet de ventiler les coûts des instances Amazon EC2 par cluster Amazon EKS individuel dans Cost Explorer. Cependant, cette identification ne capture pas les dépenses du plan de contrôle. L'identification est automatiquement ajoutée aux instances Amazon EC2 qui participent à un cluster Amazon EKS. Ce comportement se produit indépendamment du fait que les instances soient allouées à l'aide de groupes de nœuds gérés par Amazon EKS, de Karpenter ou directement avec Amazon EC2. Cette identification n'est pas prise en compte dans la limite des 50 identifications. Pour utiliser l'identification, le détenteur d'un compte doit l'activer dans la console AWS Billing ou à l'aide de l'API. Lorsqu'un propriétaire AWS Organizations de compte de gestion active le tag, celui-ci est également activé pour tous les comptes des membres de l'organisation.

Vous pouvez également organiser vos informations de facturation en fonction des ressources possédant les mêmes valeurs de clé d'étiquette. Par exemple, vous pouvez étiqueter plusieurs

ressources avec un nom d'application spécifique, puis organiser vos informations de facturation. De cette façon, vous pouvez afficher le coût total de cette application dans plusieurs services. Pour en savoir plus sur la configuration d'un rapport de répartition des coûts avec des étiquettes, consultez Rapport d'allocation des coûts mensuel dans le guide de l'utilisateur AWS Billing .

> **Note**
>
> Si vous venez d'activer la création de rapports, les données du mois en cours peuvent être consultées après 24 heures.

Cost Explorer est un outil de reporting disponible dans le cadre du niveau AWS gratuit. Vous pouvez utiliser Cost Explorer pour consulter les graphiques de vos ressources Amazon EKS des 13 derniers mois. Vous pouvez également prévoir vos dépenses sur les trois prochains mois. Vous pouvez afficher les schémas de vos dépenses en ressources AWS au fil du temps. Par exemple, vous pouvez l'utiliser pour identifier les zones qui méritent d'être approfondies et connaître les tendances que vous pouvez utiliser pour comprendre vos coûts. Vous pouvez également spécifier des plages de temps pour les données et afficher des données temporelles par jour ou par mois.

Gestion des étiquettes à l'aide de la console

À l'aide de la console Amazon EKS, vous pouvez gérer les identifications qui sont associées à des clusters et groupes de nœuds gérés nouveaux ou existants.

Lorsque vous sélectionnez une page spécifique aux ressources dans la console Amazon EKS, cette page affiche une liste de ces ressources. Par exemple, si vous sélectionnez Clusters dans le panneau de navigation de gauche, la console affiche une liste de clusters Amazon EKS. Lorsque vous sélectionnez une ressource de l'une de ces listes (par exemple, un cluster spécifique) qui prend en charge les balises, vous pouvez afficher et gérer ses balises dans l'onglet Tags (Balises).

Vous pouvez également utiliser l'éditeur de balises dans le AWS Management Console, qui fournit un moyen unifié de gérer vos balises. Pour plus d'informations, consultez la section Marquage de vos AWS ressources avec l'éditeur de balises dans le guide de l'utilisateur de l'éditeur de AWS balises.

Ajout de balises lors de la création d'une ressource

Vous pouvez ajouter des identifications aux clusters, groupes de nœuds gérés et profils Fargate Amazon EKS lorsque vous les créez. Pour plus d'informations, consultez Création d'un cluster Amazon EKS.

Ajout et suppression de balises sur une ressource

Vous pouvez ajouter ou supprimer les balises qui sont associées à vos clusters directement à partir de la page de la ressource.

Pour ajouter ou supprimer une balise sur une ressource individuelle

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans la barre de navigation, sélectionnez le Région AWS à utiliser.
3. Dans le panneau de navigation de gauche, choisissez Clusters.
4. Choisissez un cluster spécifique.
5. Choisissez l'onglet Tags (Identifications), puis Manage tags (Gérer les identifications).
6. Sur la page Gérer les balises, ajoutez ou supprimez vos balises si nécessaire.
   - Pour ajouter une identification, choisissez Ajouter une identification. Puis spécifiez la clé et la valeur de chaque balise.
   - Pour supprimer une balise, sélectionnez Remove tag (Supprimer une balise).
7. Répétez cette procédure pour chaque balise que vous voulez ajouter ou supprimer.
8. Choisissez Update (Mettre à jour) pour terminer.

Gestion des identifications à l'aide de la CLI, de l'API ou de `eksctl`

Utilisez les AWS CLI commandes ou les opérations d'API Amazon EKS suivantes pour ajouter, mettre à jour, répertorier et supprimer les balises de vos ressources. Vous pouvez uniquement utiliser `eksctl` pour ajouter des balises tout en créant simultanément les nouvelles ressources avec une seule commande.

Prise en charge de l'étiquetage pour les ressources Amazon EKS

Tâche	AWS CLI	AWS Tools for Windows PowerShell	Action d'API
Ajouter ou remplacer une ou plusieurs étiquettes.	`tag-resource`	`Add-EKSResourceTag`	`TagResource`

Tâche	AWS CLI	AWS Tools for Windows PowerShell	Action d'API
Supprimer une ou plusieurs étiquettes.	untag-resource	Remove-EKSResourceTag	UntagResource

Les exemples suivants montrent comment ajouter ou supprimer les étiquettes d'une ressource à l'aide de l' AWS CLI.

Exemple 1 : Étiqueter un cluster existant

La commande suivante permet d'étiqueter un cluster existant.

```
aws eks tag-resource --resource-arn resource_ARN --tags team=devs
```

Exemple 2 : Supprimer les étiquettes d'un cluster existant

La commande suivante permet de supprimer une étiquette d'un cluster existant.

```
aws eks untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Exemple 3 : Afficher la liste des étiquettes d'une ressource

La commande suivante permet de répertorier l'ensemble des balises qui sont associées à une ressource existante.

```
aws eks list-tags-for-resource --resource-arn resource_ARN
```

Lorsque vous utilisez certaines actions de création de ressources, vous pouvez spécifier des balises en même temps que vous créez la ressource. Les actions suivantes prennent en charge la spécification d'une balise lorsque vous créez une ressource.

Tâche	AWS CLI	AWS Tools for Windows PowerShell	Action d'API	**eksctl**
Créer un cluster	create-cluster	New-EKSCluster	CreateCluster	create cluster
Créer un groupe de nœuds géré*	create-nodegroup	New-EKSNodegroup	CreateNodegroup	create nodegroup
Créer un profil Fargate	create-fargate-profile	New-EKSFargateProfile	CreateFargateProfile.html	create fargateprofile

* Si vous souhaitez également baliser des instances Amazon EC2 lorsque vous créez un groupe de nœuds gérés, créez le groupe de nœuds gérés à l'aide d'un modèle de lancement. Pour plus d'informations, consultez Étiquetage des instances Amazon EC2. Si vos instances existent déjà, vous pouvez étiqueter manuellement les instances. Pour plus d'informations, consultez la section Marquage de vos ressources dans le guide de l'utilisateur Amazon EC2.

Service quotas Amazon EKS

Amazon EKS a intégré Service Quotas, un AWS service que vous pouvez utiliser pour consulter et gérer vos quotas depuis un emplacement central. Pour plus d'informations, veuillez consulter Qu'est-ce que Service Quotas? dans le Guide de l'utilisateur Service Quotas. Grâce à l'intégration des quotas de service, vous pouvez rapidement rechercher la valeur de votre Amazon EKS et de vos quotas AWS Fargate de service à l'aide du AWS Management Console et AWS CLI.

AWS Management Console

Pour consulter les quotas de service Amazon EKS et Fargate à l'aide du AWS Management Console

1. Ouvrez la console Service Quotas à l'adresse https://console.aws.amazon.com/servicequotas/.
2. Dans le panneau de navigation de gauche, choisissez Services AWS.

3. À partir de la liste Services AWS, recherchez et sélectionnez Amazon Elastic Kubernetes Service (Amazon EKS) ou AWS Fargate.

   Dans la liste des quotas de service, vous pouvez voir le nom du quota de service, la valeur appliquée (si elle est disponible), le quota AWS par défaut et si la valeur du quota est ajustable.
4. Pour afficher des informations supplémentaires sur un quota de service, notamment la description, choisissez le nom du quota.
5. (Facultatif) Pour demander une augmentation de quota, sélectionnez le quota que vous souhaitez augmenter, sélectionnez Request quota increase (Demander une augmentation de quota), saisissez ou sélectionnez les informations requises, puis sélectionnez Request (Demander).

Pour mieux utiliser les quotas de service à l'aide du AWS Management Console, consultez le Guide de l'utilisateur des quotas de service. Pour demander une augmentation de quota, consultez Demander une augmentation de quota dans le Guide de l'utilisateur de Service Quotas.

AWS CLI

Pour consulter les quotas de service Amazon EKS et Fargate à l'aide du AWS CLI

Exécutez la commande suivante pour afficher vos quotas Amazon EKS.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code eks \
    --output table
```

Exécutez la commande suivante pour afficher vos quotas Fargate.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

> **Note**
>
> Le quota renvoyé correspond au nombre de tâches Amazon ECS ou de Pods Amazon EKS qui peuvent être exécutés simultanément sur Fargate dans ce compte dans la Région AWS actuelle.

Pour travailler davantage avec les quotas de service à l'aide du AWS CLI, voir `service-quotas`la référence des AWS CLI commandes. Pour demander une augmentation de quota, consultez la commande `request-service-quota-increase` dans la référence des commandes AWS CLI .

Quotas de service

Nom	Par défaut	Ajusta	Description
Entrées d'accès par cluster	Chaque Région prise en charge : 3 000	Non	Le nombre maximum d'entrées d'accès par cluster.
Clusters	Chaque Région prise en charge : 100	Oui	Nombre maximal de clusters EKS pour ce compte dans la région actuelle.
Groupes de sécurité du plan de contrôle par cluster	Chaque Région prise en charge : 4	Non	Le nombre maximal de groupes de sécurité de plan de contrôle par cluster (ceux-ci sont spécifiés lorsque vous créez le cluster).
Abonnements EKS Anywhere Enterprise	Par région prise en charge : 10	Oui	Le nombre maximum d'abonnements EKS Anywhere Enterprise dans ce compte dans la région actuelle.

Nom	Par défaut	Ajusta	Description
Profils Fargate par cluster	Chaque Région prise en charge : 10	Oui	Le nombre maximal de profils Fargate par cluster.
Labéliser les pairs par sélecteur de profils Fargate	Chaque Région prise en charge : 5	Oui	Le nombre maximal de paires d'étiquettes par sélecteur de profil Fargate.
Groupes de nœuds gérés par cluster	Chaque Région prise en charge : 30	Oui	Le nombre maximal de groupes de nœuds gérés par cluster.
Nœuds par groupe de nœuds gérés	Chaque Région prise en charge : 450	Oui	Le nombre maximal de nœuds par groupe de nœuds gérés.
Plages d'adresses CIDR d'accès aux points de terminaison publics par cluster	Chaque Région prise en charge : 40	Non	Le nombre maximal de plages CIDR d'accès aux points de terminaison publics par cluster (celles-ci sont spécifiées lorsque vous créez ou mettez à jour le cluster).
Clusters enregistrés	Chaque Région prise en charge : 10	Oui	Le nombre maximal de clusters enregistrés pour ce compte dans la région actuelle.
Sélecteurs par profil Fargate	Chaque Région prise en charge : 5	Oui	Le nombre maximal de sélecteurs par profil Fargate.

> **Note**
>
> Les valeurs par défaut sont les quotas initiaux définis par AWS. Ces valeurs par défaut sont distincts des valeurs réelles de quotas appliqués et des quotas de service maximaux possible. Pour plus d'informations, veuillez consulter la rubrique Terminologie des Service Quotas dans le Guide de l'utilisateur Service Quotas.

Ces quotas de service sont répertoriés sous Amazon Elastic Kubernetes Service (Amazon EKS) dans la console Service Quotas. Pour demander une augmentation du quota pour les valeurs affichées comme ajustables, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

AWS Fargate quotas de service

Le service AWS Fargate de la console Service Quotas répertorie plusieurs quotas de service. Le tableau suivant décrit uniquement le quota qui s'applique à Amazon EKS. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour plus d'informations, consultez Création d'une alarme CloudWatch pour contrôler les métriques d'utilisation des ressources Fargate.

Les nouveaux Comptes AWS peuvent avoir des quotas initiaux plus faibles qui peuvent augmenter au fil du temps. Fargate surveille en permanence l'utilisation des comptes au sein de Région AWS chacun d'entre eux, puis augmente automatiquement les quotas en fonction de l'utilisation. Vous pouvez également demander une augmentation du quota pour les valeurs désignées comme ajustables. Pour de plus amples informations, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Nom	Par défaut	Ajustable	Description
Nombre de ressources vCPU Fargate à la demande	6	Oui	Nombre de vCPU Fargate pouvant fonctionner simultanément en tant que Fargate On-Demand dans ce compte pour la région actuelle.

> **Note**
>
> Les valeurs par défaut sont les quotas initiaux définis par AWS. Ces valeurs par défaut sont distincts des valeurs réelles de quotas appliqués et des quotas de service maximaux possible. Pour plus d'informations, veuillez consulter la rubrique Terminologie des Service Quotas dans le Guide de l'utilisateur Service Quotas.

> **Note**
>
> Fargate applique également les tâches Amazon ECS et les quotas de taux de lancement des Pods Amazon EKS. Pour plus d'informations, consultez la section relative à la AWS Fargate limitation des quotas dans le guide Amazon ECS.

Sécurité dans Amazon EKS

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit cette notion par les termes sécurité du cloud et sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. Pour Amazon EKS, AWS est responsable du plan de Kubernetes contrôle, qui inclut les nœuds du plan de contrôle et la `etcd` base de données. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de conformitéAWS. Pour en savoir plus sur les programmes de conformité qui s'appliquent à Amazon EKS, consultez Services AWS concernés par le programme de conformité.
- Sécurité dans le cloud : votre responsabilité englobe les domaines suivants :
  - La configuration de sécurité du plan de données, y compris la configuration des groupes de sécurité qui autorisent le trafic à transmettre à partir du plan de contrôle Amazon EKS dans le VPC client.
  - La configuration des nœuds et les conteneurs eux-mêmes
  - Le système d'exploitation du nœud (y compris les mises à jour et les correctifs de sécurité)
  - D'autres logiciels d'application connexes :
    - Configuration et gestion des contrôles réseau, tels que les règles de pare-feu
    - La gestion de l'identité au niveau de la plateforme et la gestion des accès, avec ou en complément de l'IAM
  - La sensibilité de vos données, les exigences de votre entreprise,et la législation et la réglementation applicables

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amazon EKS. Les rubriques suivantes vous montrent comment configurer Amazon EKS pour répondre à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources Amazon EKS.

> **Note**
>
> Les conteneurs Linux sont constitués de groupes de contrôle (cgroups) et d'espaces de noms qui aident à limiter l'accès d'un conteneur, mais tous les conteneurs partagent le même noyau Linux que l'instance Amazon EC2 hôte. L'exécution d'un conteneur en tant qu'utilisateur racine (UID 0) ou l'octroi d'un accès à un conteneur aux ressources hôtes ou aux espaces de noms tels que le réseau hôte ou l'espace de noms PID hôte sont fortement déconseillés, car cela réduit l'efficacité de l'isolation fournie par les conteneurs.

Rubriques

Signature des certificats

L'API Kubernetes Certificates automatise la mise en service des informations d'identification X.509. Cet API dispose d'une interface de ligne de commande permettant aux clients de l'API Kubernetes de demander et d'obtenir des certificats X.509 auprès d'une autorité de certification (CA). Vous pouvez utiliser la ressource `CertificateSigningRequest` (CSR) pour demander à un signataire désigné de signer le certificat. Vos demandes sont approuvées ou refusées avant d'être signées. Kubernetes prend en charge à la fois les signataires intégrés et les signataires personnalisés avec des comportements bien définis. De cette façon, les clients peuvent prédire ce qu'il advient de leurs CSR. Pour en savoir plus sur la signature des certificats, consultez les demandes de signature.

L'un des signataires intégrés est `kubernetes.io/legacy-unknown`. L'API `v1beta1` de la ressource CSR a honoré ce signataire d'héritage inconnu. Cependant, l'API `v1` stable de CSR ne permet pas de définir `signerName` sur `kubernetes.io/legacy-unknown`.

La version `1.21` et les versions antérieures d'Amazon EKS acceptaient la valeur `legacy-unknown` en tant que `signerName` dans l'API CSR `v1beta1`. Cette API permet à l'autorité de certification (CA) Amazon EKS de générer des certificats. En revanche, dans la version `1.22` de Kubernetes, l'API CSR `v1beta1` est remplacée par l'API CSR `v1`. Cette API ne prend pas en charge le signerName d'« héritage inconnu ». Si vous souhaitez utiliser Amazon EKS CA pour générer des certificats sur vos clusters, vous devez utiliser un signataire personnalisé. Il a été introduit dans la version `1.22` d'Amazon EKS. Pour utiliser la version de l'API CSR `v1` et générer un nouveau certificat, vous devez migrer tous les manifestes et clients API existants. Les certificats existants créés avec l'API `v1beta1` existante sont valides et fonctionnent jusqu'à l'expiration du certificat. Cela inclut les éléments suivants :

- Distribution d'approbation : aucune. Il n'existe pas d'approbation ou de distribution standard pour ce signataire dans un cluster Kubernetes.
- Sujets autorisés : tous
- Extensions x509 autorisées : honore subjectAltName et utilise des clés pour les extensions et supprime les autres extensions
- Utilisations de clés autorisées : ne doit pas inclure les utilisations au-delà de [« chiffrement de clé », « signature numérique », « authentification du serveur »]

> **Note**
>
> La signature de certificat client n'est pas prise en charge.

- Expiration/durée de vie du certificat : 1 an (par défaut et maximum)
- Bit CA autorisé/interdit : non autorisé

Exemple de génération CSR avec SignerName

Ces étapes montrent comment générer un certificat de service pour un nom DNS `myserver.default.svc` en utilisant `signerName: beta.eks.amazonaws.com/app-serving`. Utilisez-le comme guide pour votre propre environnement.

1. Exécutez la commande `openssl genrsa -out myserver.key 2048` pour générer une clé privée RSA.

```
openssl genrsa -out myserver.key 2048
```

2. Utilisez la commande suivante pour générer une demande de certificat.

```
openssl req -new -key myserver.key -out myserver.csr -subj "/
CN=myserver.default.svc"
```

3. Générez une valeur `base64` pour la demande CSR et stockez-la dans une variable, car vous en aurez besoin lors d'une étape ultérieure.

```
base_64=$(cat myserver.csr | base64 -w 0 | tr -d "\n")
```

4. Pour créer un fichier nommé `mycsr.yaml`, exécutez la commande suivante. Dans l'exemple suivant, `beta.eks.amazonaws.com/app-serving` est le `signerName`.

```
cat >mycsr.yaml <<EOF
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: myserver
spec:
  request: $base_64
  signerName: beta.eks.amazonaws.com/app-serving
  usages:
    - digital signature
    - key encipherment
    - server auth
EOF
```

5. Envoyez la CSR.

```
kubectl apply -f mycsr.yaml
```

6. Approuvez le certificat de service.

```
kubectl certificate approve myserver
```

7. Vérifiez que le certificat a été émis.

```
kubectl get csr myserver
```

L'exemple qui suit illustre un résultat.

```
NAME       AGE     SIGNERNAME                                  REQUESTOR
 CONDITION
myserver   3m20s   beta.eks.amazonaws.com/app-serving   kubernetes-admin
 Approved,Issued
```

8. Exportez le certificat émis.

```
kubectl get csr myserver -o jsonpath='{.status.certificate}'| base64 -d
 > myserver.crt
```

Considérations relatives à la signature des certificats avant la mise à niveau de votre cluster vers Kubernetes 1.24

Dans Kubernetes `1.23` et versions antérieures, les certificats de service `kubelet` dotés d'une adresse IP et d'un SAN (Subject Alternative Name) invérifiables étaient automatiquement émis avec un SAN invérifiable. Les SAN sont omis du certificat fourni. Dans les versions `1.24` et ultérieures des clusters, les certificats de service `kubelet` ne sont pas émis si le SAN est invérifiable. Cela entrave le fonctionnement des commandes `kubectl exec` et `kubectl logs`.

Avant de procéder à la mise à niveau de votre cluster vers `1.24`, déterminez si celui-ci possède des demandes de signature de certificat (CSR) qui n'ont pas été approuvées en effectuant les étapes suivantes :

1. Exécutez la commande suivante.

```
kubectl get csr -A
```

L'exemple qui suit illustre un résultat.

```
NAME        AGE   SIGNERNAME                        REQUESTOR
                          REQUESTEDDURATION   CONDITION
```

```
csr-7znmf   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-42-149.region.compute.internal      <none>
 Approved
csr-9xx5q   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-65-38.region.compute.internal      <none>
 Approved, Issued
```

Si la sortie renvoyée présente un CSR dont le signataire `kubernetes.io/kubelet-serving` est `Approved` mais pas `Issued` pour un nœud, vous devez approuver la demande.

2. Approuvez manuellement le CSR. Remplacez `csr-7znmf` par votre propre valeur.

```
kubectl certificate approve csr-7znmf
```

À l'avenir, pour approuver automatiquement les CSR, nous vous recommandons de créer un contrôleur d'approbation capable de valider et d'approuver automatiquement les CSR contenant des adresses IP ou des SAN DNS qu'Amazon EKS n'a pas la capacité de vérifier.

Gestion des identités et des accès pour Amazon EKS

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) pour utiliser des ressources Amazon EKS. 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 EKS.

Utilisateur du service : si vous utilisez le service Amazon EKS pour accomplir votre tâche, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Vous pourrez avoir besoin d'autorisations supplémentaires si vous utilisez davantage de fonctionnalités Amazon EKS. 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 EKS, consultez Dépannage IAM.

Administrateur du service : Si vous êtes le responsable des ressources Amazon EKS de votre entreprise, vous bénéficiez probablement d'un accès total à ce service. C'est à vous que revient de déterminer les fonctions et les ressources Amazon EKS 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 en savoir plus sur la façon dont votre entreprise peut utiliser IAM avec Amazon EKS, consultez Fonctionnement d'Amazon EKS avec IAM.

Administrateur IAM : Si vous êtes un administrateur IAM, vous souhaiterez peut-être obtenir des informations sur la façon dont vous pouvez écrire des politiques pour gérer l'accès à Amazon EKS. Pour afficher des exemples de politiques basées sur l'identité Amazon EKS que vous pouvez utiliser dans IAM, consultez Exemples de politiques basées sur l'identité d'Amazon EKS.

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, veuillez consulter Multi-factor authentication (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 les informations d'identification de l'utilisateur root dans le Guide de l'utilisateur IAM.

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, veuillez consulter la rubrique Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte – Vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, une fonction 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.
- Fonction du service – Il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.
- Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance EC2 et qui envoient des demandes d'API. AWS CLI AWS Cette solution est préférable au stockage des clés d'accès au sein de l'instance EC2. Pour attribuer un AWS rôle à une instance EC2 et le mettre à la disposition de toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance EC2 d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur IAM.

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 Présentation 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 Amazon S3. 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 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.

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 Présentation 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 fonction 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 qui en résultent 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 fonctions 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 passez en tant que paramètre lorsque vous programmez afin de créer une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de la séance obtenue 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 remplace 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 obtenues 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.

Fonctionnement d'Amazon EKS avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon EKS, vous devez comprendre quelles sont les fonctionnalités IAM qui peuvent être utilisées dans cette situation. Pour obtenir une vue d'ensemble de la manière dont Amazon EKS et les autres AWS services fonctionnent avec IAM, consultez la section AWS Services compatibles avec IAM dans le guide de l'utilisateur d'IAM.

Rubriques

- Politiques basées sur l'identité Amazon EKS
- Politiques basées sur les ressources Amazon EKS
- Autorisation basée sur des identifications Amazon EKS
- Rôles IAM Amazon EKS

Politiques basées sur l'identité Amazon EKS

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. Amazon EKS est compatible avec des actions, des ressources et des clés de condition spécifiques. Pour en savoir plus sur 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.

Actions

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 politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Les actions de politique dans Amazon EKS utilisent le préfixe suivant avant l'action : `eks:`. Par exemple, pour accorder à une personne l'autorisation d'obtenir des informations descriptives sur un cluster Amazon EKS, incluez l'action `DescribeCluster` dans sa politique. Les déclarations de politique doivent inclure un élément `Action` ou `NotAction`.

Pour spécifier plusieurs actions dans une seule déclaration, séparez-les par des virgules comme suit :

```
"Action": ["eks:action1", "eks:action2"]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `Describe`, incluez l'action suivante :

```
"Action": "eks:Describe*"
```

Pour afficher la liste des actions Amazon EKS, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service de la Référence de l'autorisation de service.

Ressources

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 pour lesquels 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": "*"
```

La ressource de cluster Amazon EKS intègre l'ARN suivant.

```
arn:aws:eks:region-code:account-id:cluster/cluster-name
```

Pour plus d'informations sur le format des ARN, consultez Amazon resource names (ARN) and AWS service namespaces.

Par exemple, pour spécifier le cluster dont le nom figure *my-cluster* dans votre instruction, utilisez l'ARN suivant :

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/my-cluster"
```

Pour spécifier tous les clusters appartenant à un compte spécifique et Région AWS utiliser le caractère générique (*) :

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/*"
```

Certaines actions Amazon EKS, telles que celles destinées à la création de ressources, ne peuvent pas être exécutées sur une ressource spécifique. Dans ces cas-là, vous devez utiliser le caractère générique (*).

```
"Resource": "*"
```

Pour afficher la liste des types de ressources Amazon EKS et leurs ARN, consultez la rubrique Ressources définies par Amazon Elastic Kubernetes Service de la Référence de l'autorisation de service. Pour connaître les actions avec lesquelles vous pouvez spécifier l'ARN de chaque ressource, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service.

Clés de condition

Amazon EKS définit son propre ensemble de clés de condition et est également compatible avec l'utilisation de certaines clés de condition globales. Pour voir toutes les clés de condition AWS globales, consultez la section Clés contextuelles de condition AWS globale dans le guide de l'utilisateur IAM.

Vous pouvez définir des clés de condition lors de l'association d'un fournisseur OpenID Connect à votre cluster. Pour plus d'informations, consultez Exemple de politique IAM.

Toutes les actions Amazon EC2 prennent en charge les clés de condition `aws:RequestedRegion` et `ec2:Region`. Pour plus d'informations, voir Exemple : restriction de l'accès à un élément spécifique Région AWS.

Pour obtenir la liste des clés de condition Amazon EKS, consultez la rubrique Conditions définies par Amazon Elastic Kubernetes Service de la Référence de l'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service.

Exemples

Pour voir des exemples de politiques Amazon EKS basées sur l'identité, consultez Exemples de politiques basées sur l'identité d'Amazon EKS.

Lorsque vous créez un cluster Amazon EKS, le principal IAM qui crée le cluster se voit automatiquement accorder des autorisations `system:masters` dans la configuration du contrôle d'accès basé sur les rôles (RBAC) du cluster dans le plan de contrôle Amazon EKS. Ce principal n'apparaît dans aucune configuration visible. Souvenez-vous donc du principal qui a créé le cluster

à l'origine. Pour permettre à d'autres principaux IAM d'interagir avec votre cluster, modifiez la `ConfigMap aws-auth` dans Kubernetes et créez un `rolebinding` ou un `clusterrolebinding` Kubernetes avec le nom d'un `group` que vous spécifiez dans la `ConfigMap aws-auth`.

Pour plus d'informations sur l'utilisation du ConfigMap, consultezAutoriser l'accès aux Kubernetes API .

Politiques basées sur les ressources Amazon EKS

Amazon EKS ne prend pas en charge les politiques basées sur les ressources.

Autorisation basée sur des identifications Amazon EKS

Vous pouvez attacher des identifications aux ressources Amazon EKS ou transmettre des identifications dans une demande à Amazon EKS. Pour contrôler l'accès basé sur des identifications, vous devez fournir les informations d'identifications dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Pour plus d'informations sur l'étiquetage des ressources Amazon EKS, consultez Étiquetage de vos ressources Amazon EKS. Pour plus d'informations sur les actions dans lesquelles vous pouvez utiliser des balises avec des clés de condition, consultez Actions définies par Amazon EKS dans Référence de l'autorisation de service.

Rôles IAM Amazon EKS

Un rôle IAM est une entité de votre AWS compte qui possède des autorisations spécifiques.

Utilisation d'informations d'identification temporaires avec Amazon EKS

Vous pouvez utiliser des informations d'identification temporaires pour vous connecter à l'aide de la fédération, endosser un rôle IAM ou encore pour endosser un rôle intercompte. Vous obtenez des informations d'identification de sécurité temporaires en appelant des opérations d' AWS STS API telles que AssumeRoleou GetFederationToken.

Amazon EKS est compatible avec l'utilisation des informations d'identification temporaires.

Rôles liés à un service

Les rôles liés aux AWS services permettent aux services d'accéder aux ressources d'autres services pour effectuer une action en votre nom. Les rôles liés à un service s'affichent dans votre compte IAM et sont la propriété du service. Un administrateur peut afficher, mais ne peut pas modifier les autorisations concernant les rôles liés à un service.

Amazon EKS prend en charge les rôles liés à un service. Pour plus d'informations sur la création ou la gestion des rôles liés à un service Amazon EKS, consultez Utilisation des rôles liés à un service pour Amazon EKS.

Rôles de service

Cette fonction permet à un service d'endosser une fonction du service en votre nom. Ce rôle autorise le service à accéder à des ressources d'autres services pour effectuer une action en votre nom. Les fonctions du service s'affichent dans votre compte IAM et sont la propriété du compte. Cela signifie qu'un administrateur IAM peut modifier les autorisations associées à ce rôle. Toutefois, une telle action peut perturber le bon fonctionnement du service.

Amazon EKS prend en charge les rôles de service. Pour plus d'informations, consultez Rôle IAM de cluster Amazon EKS et Rôle IAM de nœud Amazon EKS.

Choix d'un rôle IAM dans Amazon EKS

Lorsque vous créez une ressource de cluster dans Amazon EKS, vous devez choisir un rôle pour permettre à Amazon EKS d'accéder à plusieurs autres AWS ressources en votre nom. Si vous avez déjà créé un rôle de service, Amazon EKS vous propose une liste de rôles parmi lesquels choisir. Il est important de choisir un rôle auquel les politiques gérées par Amazon EKS sont attachées. Pour plus d'informations, consultez Recherche d'un rôle de cluster existant et Recherche d'un rôle de nœud existant.

Exemples de politiques basées sur l'identité d'Amazon EKS

Par défaut, les utilisateurs et les rôles IAM ne sont pas autorisés à créer ou modifier les ressources Amazon EKS. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l' AWS API AWS Management Console AWS CLI, ou. Un administrateur IAM doit créer des politiques IAM autorisant les utilisateurs et les rôles à exécuter des opérations d'API spécifiques sur les ressources spécifiées dont ils ont besoin. Il doit ensuite attacher ces politiques aux utilisateurs ou aux groupes IAM ayant besoin de ces autorisations.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, veuillez consulter Création de politiques dans l'onglet JSON dans le Guide de l'utilisateur IAM.

Lorsque vous créez un cluster Amazon EKS, le principal IAM qui crée le cluster se voit automatiquement accorder des autorisations `system:masters` dans la configuration du contrôle

d'accès basé sur les rôles (RBAC) du cluster dans le plan de contrôle Amazon EKS. Ce principal n'apparaît dans aucune configuration visible. Souvenez-vous donc du principal qui a créé le cluster à l'origine. Pour permettre à d'autres principaux IAM d'interagir avec votre cluster, modifiez la `ConfigMap aws-auth` dans Kubernetes et créez un `rolebinding` ou un `clusterrolebinding` Kubernetes avec le nom d'un `group` que vous spécifiez dans la `ConfigMap aws-auth`.

Pour plus d'informations sur l'utilisation du ConfigMap, consultezAutoriser l'accès aux Kubernetes API .

Rubriques

- Bonnes pratiques en matière de politiques
- Utilisation de la console Amazon EKS
- Autorisation des utilisateurs IAM à afficher leurs propres autorisations
- Créer un cluster Kubernetes sur le AWS Cloud
- Créer un cluster Kubernetes local sur un Outpost
- Mettre à jour un cluster Kubernetes
- Affichage de la liste ou description de tous les clusters

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si une personne peut créer, consulter ou supprimer des ressources Amazon EKS 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 les AWS clients spécifiques à vos cas d'utilisation. Pour de plus amples informations, consultez Politiques gérées AWS ou Politiques gérées 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 de plus amples 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 de plus amples 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 EKS

Pour accéder à la console Amazon EKS, un principal IAM doit disposer d'un ensemble minimum d'autorisations. Ces autorisations permettent au principal de répertorier et de consulter les informations relatives aux ressources Amazon EKS de votre AWS compte. Si vous créez une politique basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les principaux auxquels cette politique est attachée.

Pour garantir que vos principaux IAM puissent toujours utiliser la console Amazon EKS, créez une politique avec votre propre nom unique, par exemple `AmazonEKSAdminPolicy`. Attachez la politique aux principaux. Pour plus d'informations, consultez la rubrique Ajout et suppression d'autorisations basées sur l'identité IAM du Guide de l'utilisateur IAM.

> **Important**
>
> L'exemple de politique suivant permet à un principal d'afficher des informations dans l'onglet Configuration sur la console. Pour afficher des informations sur les onglets Présentation et Ressources dans la AWS Management Console, le principal a également besoin des autorisations Kubernetes. Pour plus d'informations, consultez Autorisations nécessaires.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux principaux 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 que vous tentez d'effectuer.

Autorisation des utilisateurs IAM à 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": "*"
        }
    ]
}
```

Créer un cluster Kubernetes sur le AWS Cloud

Région AWS Vous pouvez le Région AWS remplacer par Région AWS celui dans lequel vous souhaitez créer un cluster. Si vous voyez un avertissement indiquant Les actions de votre politique ne prennent pas en charge les autorisations au niveau des ressources :**`All resources`**dans le AWS Management Console, vous pouvez l'ignorer sans risque. Si le *`AWSServiceRoleForAmazonEKS`*rôle est déjà attribué à votre compte, vous pouvez supprimer l'`iam:CreateServiceLinkedRole`action de la politique. Si vous avez déjà créé un cluster Amazon EKS dans votre compte, ce rôle existe déjà, sauf si vous l'avez supprimé.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
eks.amazonaws.com/AWSServiceRoleForAmazonEKS",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iam:AWSServiceName": "eks"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        }
    ]
}
```

Créer un cluster Kubernetes local sur un Outpost

Région AWS Vous pouvez le Région AWS remplacer par Région AWS celui dans lequel vous souhaitez créer un cluster. Si vous voyez un avertissement indiquant Les actions de votre politique ne prennent pas en charge les autorisations au niveau des ressources :**`All resources`**dans le AWS Management Console, vous pouvez l'ignorer sans risque. Si votre compte possède déjà le rôle `AWSServiceRoleForAmazonEKSLocalOutpost`, vous pouvez supprimer l'action `iam:CreateServiceLinkedRole` de la politique. Si vous avez déjà créé un cluster local Amazon EKS sur un Outpost dans votre compte, ce rôle existe déjà, sauf si vous l'avez supprimé.

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

```
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "iam:GetRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/outposts.eks-
local.amazonaws.com/AWSServiceRoleForAmazonEKSLocalOutpost"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "iam:ListAttachedRolePolicies"
            ]
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/eks-local-*",
            "Effect": "Allow"
        },
    ]
}
```

Mettre à jour un cluster Kubernetes

Région AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:UpdateClusterVersion",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        }
    ]
}
```

Affichage de la liste ou description de tous les clusters

Cet exemple de politique inclut les autorisations minimales requises pour répertorier et décrire tous les clusters de votre compte. Un principal IAM doit être en mesure de répertorier et de décrire les clusters pour utiliser la `update-kubeconfig` AWS CLI commande.

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

Utilisation des rôles liés à un service pour Amazon EKS

Amazon Elastic Kubernetes Service AWS Identity and Access Management utilise des rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon EKS. Les rôles liés à un service sont prédéfinis par Amazon EKS et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Rubriques

Utilisation de rôles pour les clusters Amazon EKS

Amazon Elastic Kubernetes Service AWS Identity and Access Management utilise des rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon EKS. Les rôles liés à un service sont prédéfinis par Amazon EKS et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon EKS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon EKS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon EKS peut endosser ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon EKS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Yes (oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations du rôle lié à un service pour Amazon EKS

Amazon EKS utilise le rôle lié à un service nommé `AWSServiceRoleForAmazonEKS` : le rôle permet à Amazon EKS de gérer des clusters dans votre compte. Les politiques attachées permettent au rôle de gérer les ressources suivantes : interfaces réseau, groupes de sécurité, journaux et VPC.

> **Note**
>
> Le rôle lié à un service `AWSServiceRoleForAmazonEKS` est distinct du rôle requis pour la création de cluster. Pour plus d'informations, consultez Rôle IAM de cluster Amazon EKS.

Le rôle lié à un service `AWSServiceRoleForAmazonEKS` approuve les services suivants pour endosser le rôle :

- `eks.amazonaws.com`

La politique d'autorisations liée au rôle permet à Amazon EKS de réaliser les actions suivantes sur les ressources spécifiées :

- `AmazonEKSServiceRolePolicy`

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon EKS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un cluster dans le AWS Management Console, le ou l' AWS API AWS CLI, Amazon EKS crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un cluster, Amazon EKS crée automatiquement le rôle lié au service à nouveau.

Modification d'un rôle lié à un service pour Amazon EKS

Amazon EKS ne vous autorise pas à modifier le rôle lié à un service `AWSServiceRoleForAmazonEKS`. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Amazon EKS

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer manuellement.

Nettoyer un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez supprimer toutes les ressources utilisées par le rôle.

> **Note**
>
> Si le service Amazon EKS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon EKS utilisées par le rôle **`AWSServiceRoleForAmazonEKS`**

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Si votre cluster possède des groupes de nœuds ou des profils Fargate, vous devez les supprimer avant de pouvoir supprimer le cluster. Pour plus d'informations, consultez Suppression d'un groupe de nœuds gérés et Suppression d'un profil Fargate.
4. Dans la page Clusters, choisissez le cluster à supprimer et cliquez sur Delete (Supprimer).
5. Tapez le nom du cluster dans la fenêtre de confirmation de suppression, puis choisissez Delete (Supprimer).
6. Répétez cette procédure pour tous les autres clusters de votre compte. Attendez la fin de toutes les opérations de suppression.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au `AWSServiceRoleForAmazonEKS` service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Amazon EKS

Amazon EKS prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Amazon EKS endpoints and quotas (langue française non garantie).

Utilisation de rôles pour les groupes de nœuds Amazon EKS

Amazon EKS utilise des AWS Identity and Access Management rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon EKS. Les rôles liés à un service sont prédéfinis par Amazon EKS et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon EKS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon EKS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon EKS peut endosser ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon EKS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Yes (oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations du rôle lié à un service pour Amazon EKS

Amazon EKS utilise le rôle lié à un service nommé `AWSServiceRoleForAmazonEKSNodegroup` : le rôle permet à Amazon EKS de gérer les groupes de nœuds dans votre compte. Les politiques attachées permettent au rôle de gérer les ressources suivantes : groupes Auto Scaling, groupes de sécurité, modèles de lancement et profils d'instance IAM.

Le rôle lié à un service `AWSServiceRoleForAmazonEKSNodegroup` approuve les services suivants pour endosser le rôle :

- `eks-nodegroup.amazonaws.com`

La politique d'autorisations liée au rôle permet à Amazon EKS de réaliser les actions suivantes sur les ressources spécifiées :

- AWSServiceRoleForAmazonEKSNodegroup

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon EKS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous CreateNodegroup utilisez l'API AWS Management Console AWS CLI, le ou l' AWS API, Amazon EKS crée le rôle lié au service pour vous.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Si vous utilisiez le service Amazon EKS avant le 1er janvier 2017, date à laquelle il a commencé à prendre en charge les rôles liés au service, Amazon EKS a créé le AWSServiceRoleForAmazonEKSNodegroup rôle dans votre compte. Pour en savoir plus, consultez Un nouveau rôle est apparu dans mon compte IAM.

Création d'un rôle lié à un service dans Amazon EKS (API)AWS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un groupe de nœuds gérés dans le AWS Management Console AWS CLI, le ou l' AWS API, Amazon EKS crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un autre groupe de nœuds gérés, Amazon EKS crée automatiquement le rôle lié au service.

Modification d'un rôle lié à un service pour Amazon EKS

Amazon EKS ne vous autorise pas à modifier le rôle lié à un service `AWSServiceRoleForAmazonEKSNodegroup`. Une fois que vous avez créé un rôle lié à un service,

vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Amazon EKS

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer manuellement.

Nettoyer un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez supprimer toutes les ressources utilisées par le rôle.

> **Note**
>
> Si le service Amazon EKS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon EKS utilisées par le rôle **`AWSServiceRoleForAmazonEKSNodegroup`**

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Sélectionnez l'onglet Compute (Calcul).
4. Dans la section Node Groups (Groupes de nœuds), choisissez le groupe de nœuds à supprimer.
5. Tapez le nom du groupe de nœuds dans la fenêtre de confirmation de suppression, puis choisissez Delete (Supprimer).
6. Répétez cette procédure pour tous les autres groupes de nœuds du cluster. Attendez la fin de toutes les opérations de suppression.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au `AWSServiceRoleForAmazonEKSNodegroup` service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Amazon EKS

Amazon EKS prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Amazon EKS endpoints and quotas (langue française non garantie).

Utilisation de rôles pour les profils Amazon EKS Fargate

Amazon EKS utilise des AWS Identity and Access Management rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon EKS. Les rôles liés à un service sont prédéfinis par Amazon EKS et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon EKS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon EKS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon EKS peut endosser ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon EKS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Yes (oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations du rôle lié à un service pour Amazon EKS

Amazon EKS utilise le rôle lié à un service nommé `AWSServiceRoleForAmazonEKSForFargate` : le rôle permet à Amazon EKS Fargate de configurer les réseaux VPC requis pour les Pods Fargate. Les politiques attachées permettent au rôle de créer et de supprimer des interfaces réseau Elastic et de décrire les interfaces et les ressources réseau Elastic.

Le rôle lié à un service `AWSServiceRoleForAmazonEKSForFargate` approuve les services suivants pour endosser le rôle :

- `eks-fargate.amazonaws.com`

La politique d'autorisations liée au rôle permet à Amazon EKS de réaliser les actions suivantes sur les ressources spécifiées :

- `AmazonEKSForFargateServiceRolePolicy`

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon EKS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un profil Fargate dans l'API, dans AWS Management Console le ou dans AWS CLI l' AWS API, Amazon EKS crée le rôle lié au service pour vous.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Si vous utilisiez le service Amazon EKS avant le 13 décembre 2019, date à laquelle il a commencé à prendre en charge les rôles liés au service, Amazon EKS a créé le AWSServiceRoleForAmazonEKSForFargate rôle dans votre compte. Pour plus d'informations, consultez Un nouveau rôle est apparu dans mon compte IAM.

Création d'un rôle lié à un service dans Amazon EKS (API)AWS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un profil Fargate dans l'API, dans AWS Management Console le ou dans AWS CLI l' AWS API, Amazon EKS crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un autre groupe de nœuds gérés, Amazon EKS crée automatiquement le rôle lié au service.

Modification d'un rôle lié à un service pour Amazon EKS

Amazon EKS ne vous autorise pas à modifier le rôle lié à un service `AWSServiceRoleForAmazonEKSForFargate`. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Amazon EKS

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer manuellement.

Nettoyer un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez supprimer toutes les ressources utilisées par le rôle.

> **Note**
>
> Si le service Amazon EKS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon EKS utilisées par le rôle **`AWSServiceRoleForAmazonEKSForFargate`**

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Sur la page Clusters, sélectionnez votre cluster.
4. Sélectionnez l'onglet Compute (Calcul).
5. S'il existe des profils Fargate dans la section Profils Fargate, sélectionnez chacun individuellement, puis choisissez Delete (Supprimer).
6. Tapez le nom du profile dans la fenêtre de confirmation de suppression, puis choisissez Delete (Supprimer).

7. Répétez cette procédure pour tous les autres profils Fargate dans le cluster et pour tous les autres clusters de votre compte.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au AWSServiceRoleForAmazonEKSForFargate service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Amazon EKS

Amazon EKS prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Amazon EKS endpoints and quotas (langue française non garantie).

Utilisation de rôles pour connecter un cluster Kubernetes à Amazon EKS

Amazon EKS utilise des AWS Identity and Access Management rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon EKS. Les rôles liés à un service sont prédéfinis par Amazon EKS et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon EKS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon EKS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon EKS peut endosser ses rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon EKS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Yes (oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations du rôle lié à un service pour Amazon EKS

Amazon EKS utilise le rôle lié à un service nommé `AWSServiceRoleForAmazonEKSConnector` : le rôle permet à Amazon EKS de connecter des clusters Kubernetes. Les politiques attachées

permettent au rôle de gérer les ressources nécessaires pour se connecter à votre cluster Kubernetes enregistré.

Le rôle lié à un service `AWSServiceRoleForAmazonEKSConnector` approuve les services suivants pour endosser le rôle :

- `eks-connector.amazonaws.com`

La politique d'autorisations liée au rôle permet à Amazon EKS de réaliser les actions suivantes sur les ressources spécifiées :

- `AmazonEKSConnectorServiceRolePolicy`

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon EKS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service pour connecter un cluster. Lorsque vous connectez un cluster dans le AWS Management Console, le AWS CLI`eksctl`, ou l'AWS API, Amazon EKS crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous connectez un cluster, Amazon EKS crée à nouveau le rôle lié à un service pour vous.

Modification d'un rôle lié à un service pour Amazon EKS

Amazon EKS ne vous autorise pas à modifier le rôle lié à un service `AWSServiceRoleForAmazonEKSConnector`. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Amazon EKS

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée

qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer manuellement.

Nettoyer un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez supprimer toutes les ressources utilisées par le rôle.

> **Note**
>
> Si le service Amazon EKS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon EKS utilisées par le rôle **`AWSServiceRoleForAmazonEKSConnector`**

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans le panneau de navigation de gauche, choisissez Clusters.
3. Sur la page Clusters, sélectionnez votre cluster.
4. Sélectionnez les onglets Annuler l'inscription puis Ok.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au AWSServiceRoleForAmazonEKSConnector service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Utilisation de rôles pour les clusters locaux Amazon EKS sur Outpost

Amazon Elastic Kubernetes Service AWS Identity and Access Management utilise des rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon EKS. Les rôles liés à un service sont prédéfinis par Amazon EKS et incluent toutes les autorisations requises par le service pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon EKS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon EKS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon EKS peut endosser ses rôles. Les autorisations

définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon EKS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Yes (oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations du rôle lié à un service pour Amazon EKS

Amazon EKS utilise le rôle lié à un service nommé `AWSServiceRoleForAmazonEKSLocalOutpost` : le rôle permet à Amazon EKS de gérer des clusters locaux dans votre compte. Les politiques attachées permettent au rôle de gérer les ressources suivantes : interfaces réseau, groupes de sécurité, journaux et instances Amazon EC2.

> **Note**
>
> Le rôle lié à un service `AWSServiceRoleForAmazonEKSLocalOutpost` est distinct du rôle requis pour la création de cluster. Pour plus d'informations, consultez Rôle IAM de cluster Amazon EKS.

Le rôle lié à un service `AWSServiceRoleForAmazonEKSLocalOutpost` approuve les services suivants pour endosser le rôle :

- `outposts.eks-local.amazonaws.com`

La politique d'autorisations liée au rôle permet à Amazon EKS de réaliser les actions suivantes sur les ressources spécifiées :

- `AmazonEKSServiceRolePolicy`

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon EKS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un cluster dans le AWS Management Console, le ou l' AWS API AWS CLI, Amazon EKS crée le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un cluster, Amazon EKS crée automatiquement le rôle lié au service à nouveau.

Modification d'un rôle lié à un service pour Amazon EKS

Amazon EKS ne vous autorise pas à modifier le rôle lié à un service `AWSServiceRoleForAmazonEKSLocalOutpost`. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence au rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour en savoir plus, consultez Modification d'un rôle lié à un service dans le guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Amazon EKS

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer manuellement.

Nettoyer un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez supprimer toutes les ressources utilisées par le rôle.

> **Note**
>
> Si le service Amazon EKS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon EKS utilisées par le rôle **`AWSServiceRoleForAmazonEKSLocalOutpost`**

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

2. Sélectionnez Amazon EKS Clusters (Clusters Amazon EKS) dans le panneau de navigation de gauche.
3. Si votre cluster possède des groupes de nœuds ou des profils Fargate, vous devez les supprimer avant de pouvoir supprimer le cluster. Pour plus d'informations, consultez Suppression d'un groupe de nœuds gérés et Suppression d'un profil Fargate.
4. Dans la page Clusters, choisissez le cluster à supprimer et cliquez sur Delete (Supprimer).
5. Tapez le nom du cluster dans la fenêtre de confirmation de suppression, puis choisissez Delete (Supprimer).
6. Répétez cette procédure pour tous les autres clusters de votre compte. Attendez la fin de toutes les opérations de suppression.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au `AWSServiceRoleForAmazonEKSLocalOutpost` service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Amazon EKS

Amazon EKS prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Amazon EKS endpoints and quotas (langue française non garantie).

Rôle IAM de cluster Amazon EKS

Le rôle IAM du cluster Amazon EKS est requis pour chaque cluster. Les clusters Kubernetes gérés par Amazon EKS utilisent ce rôle pour gérer les nœuds et le fournisseur de cloud existant utilise ce rôle pour créer des équilibreurs de charge avec Elastic Load Balancing pour les services.

Avant de pouvoir créer des clusters Amazon EKS, vous devez créer un rôle IAM avec l'une des politiques IAM suivantes :

- `AmazonEKSClusterPolicy`
- Une politique IAM personnalisée. Les autorisations minimales suivantes permettent au cluster Kubernetes de gérer les nœuds, mais n'autorisent pas le fournisseur de cloud existant à créer des équilibreurs de charge avec Elastic Load Balancing. Votre politique IAM personnalisée doit disposer au moins des autorisations suivantes :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:*:*:instance/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:TagKeys": "kubernetes.io/cluster/*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeAvailabilityZones",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

Note

Avant le 3 octobre 2023, le rôle IAM d'ClusterPolicyAmazoneks était requis pour chaque cluster.
Avant le 16 avril 2020, Amazoneks et ServicePolicy ClusterPolicy Amazoneks étaient obligatoires et le nom suggéré pour le rôle était. `eksServiceRole` Avec le rôle `AWSServiceRoleForAmazonEKS` lié à un service, la ServicePolicy politique Amazoneks n'est plus requise pour les clusters créés le 16 avril 2020 ou après cette date.

Recherche d'un rôle de cluster existant

Vous pouvez utiliser la procédure suivante pour vérifier si votre compte possède déjà le rôle de cluster Amazon EKS.

Pour vérifier **`eksClusterRole`** dans la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Recherchez dans la liste des rôles `eksClusterRole`. Si un rôle qui inclut `eksClusterRole` n'existe pas, consultez Création du rôle de cluster Amazon EKS pour créer le rôle. Si un rôle qui inclut `eksClusterRole` existe, sélectionnez le rôle pour afficher les politiques attachées.
4. Choisissez Permissions (Autorisations).
5. Assurez-vous que la politique gérée ClusterPolicy par Amazoneks est attachée au rôle. Si la politique est attachée, votre rôle de cluster Amazon EKS est configuré correctement.
6. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la relation d'approbation).
7. Vérifiez que la relation d'approbation contient la politique suivante. Si la relation d'approbation correspond à la politique ci-dessous, sélectionnez Annuler. Si la relation d'approbation ne correspond pas, copiez la politique dans la fenêtre Edit trust policy (Modifier la politique d'approbation) et sélectionnez Update policy (Mettre à jour la politique).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Création du rôle de cluster Amazon EKS

Vous pouvez utiliser le AWS Management Console ou le AWS CLI pour créer le rôle de cluster.

AWS Management Console

Pour créer votre rôle de cluster Amazon EKS dans la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Roles (Rôles) puis Create role (Créer un rôle).
3. Sous Type d'entité approuvée, choisissez Service AWS .
4. Dans la liste déroulante Use cases for other Services AWS (Cas d'utilisation pour d'autres), sélectionnez EKS.
5. Sélectionnez EKS - Cluster pour votre cas d'utilisation, puis Next (Suivant).
6. Sous l'onglet Add permissions (Ajouter des autorisations), sélectionnez Next (Suivant).
7. Pour Role name (Nom de rôle), saisissez un nom unique pour votre rôle, par exemple, **`eksClusterRole`**.
8. Pour Description, saisissez un texte descriptif tel que **`Amazon EKS - Cluster role`**.
9. Sélectionnez Créer un rôle.

AWS CLI

1. Copiez le contenu suivant dans un fichier nommé *`cluster-trust-policy.json`*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Créez le rôle. Vous pouvez remplacer ***`eksClusterRole`*** par n'importe quel nom que vous choisissez.

```
aws iam create-role \
  --role-name eksClusterRole \
```

```
  --assume-role-policy-document file://"cluster-trust-policy.json"
```

3. Attachez la politique IAM requise au rôle.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name eksClusterRole
```

Rôle IAM de nœud Amazon EKS

Le `kubelet` démon du nœud Amazon EKS passe des appels aux AWS API en votre nom. Les nœuds reçoivent l'autorisation pour ces appels d'API via un profil d'instance IAM et les politiques associées. Avant de pouvoir lancer les nœuds et les enregistrer dans un cluster, vous devez créer un rôle IAM qui sera utilisé par ces nœuds lors de leur lancement. Cette exigence s'applique aux nœuds lancés avec l'AMI optimisée pour Amazon EKS fournie par Amazon, ou avec toute autre AMI de nœud que vous prévoyez d'utiliser. En outre, cette exigence s'applique à la fois aux groupes de nœuds gérés et aux nœuds autogérés.

> **Note**
>
> Vous ne pouvez pas utiliser le même rôle que celui utilisé pour créer des clusters.

Avant de créer des nœuds, vous devez créer un rôle IAM avec les autorisations suivantes :

- Autorisations permettant au `kubelet` de décrire les ressources Amazon EC2 dans le VPC, telles que fournies par la politique `AmazonEKSWorkerNodePolicy`. Cette politique fournit également les autorisations pour l'agent d'identité du pod Amazon EKS.
- Autorisations permettant au `kubelet` d'utiliser des images du conteneur provenant d'Amazon Elastic Container Registry (Amazon ECR), telles que prévues par la politique `AmazonEC2ContainerRegistryReadOnly`. Les autorisations d'utiliser des images de conteneurs provenant d'Amazon Elastic Container Registry (Amazon ECR) sont nécessaires parce que les modules complémentaires intégrés pour la mise en réseau exécutent des pods qui utilisent des images de conteneurs provenant d'Amazon ECR.
- (Facultatif) Autorisations permettant à l'agent d'identité du pod Amazon EKS d'utiliser l'action `eks-auth:AssumeRoleForPodIdentity` pour récupérer les informations d'identification pour les

pods. Si vous n'utilisez pas WorkerNodePolicyAmazoneKS, vous devez fournir cette autorisation en plus des autorisations EC2 pour utiliser EKS Pod Identity.

- (Facultatif) Si vous n'utilisez pas IRSA ou l'identité du pod EKS pour accorder des autorisations aux pods CNI VPC, vous devez fournir des autorisations pour le CNI VPC sur le rôle d'instance. Vous pouvez utiliser soit la politique gérée par `AmazonEKS_CNI_Policy` (si vous avez créé votre cluster avec la famille `IPv4`), soit une politique IPv6 que vous créez (si vous avez créé votre cluster avec la famille `IPv6`). Cependant, plutôt que d'attacher la politique à ce rôle, nous vous recommandons de l'attacher à un rôle distinct utilisé spécifiquement pour le module complémentaire Amazon VPC CNI. Pour plus d'informations sur la création d'un rôle distinct pour le module complémentaire Amazon VPC CNI, veuillez consulter Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

> **Note**
>
> Avant le 3 octobre 2023, `AmazonEKSWorkerNodePolicy` et `AmazonEC2ContainerRegistryReadOnly` étaient requises sur le rôle IAM pour chaque groupe de nœuds gérés.
> Les groupes de nœuds Amazon EC2 doivent avoir un rôle IAM différent de celui du profil Fargate. Pour plus d'informations, consultez Rôle IAM d'exécution de Pod Amazon EKS.

Recherche d'un rôle de nœud existant

Vous pouvez utiliser la procédure suivante pour vérifier si votre compte possède déjà le rôle de nœud Amazon EKS.

Pour vérifier **`eksNodeRole`** dans la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Recherchez `eksNodeRole`, `AmazonEKSNodeRole` ou `NodeInstanceRole` dans la liste des rôles. Si aucun rôle portant l'un de ces noms n'existe, consultez la section Création du rôle IAM de nœud Amazon EKS pour créer le rôle. Si un rôle qui contient `eksNodeRole`, `AmazonEKSNodeRole` ou `NodeInstanceRole` existe, sélectionnez-le pour afficher les politiques attachées.
4. Choisissez Autorisations.

5. Assurez-vous que les politiques gérées par WorkerNodePolicy Amazoneks et ContainerRegistryReadOnly AmazonEC2 sont associées au rôle ou qu'une politique personnalisée est attachée avec les autorisations minimales.

> **Note**
>
> Si la politique AmazonEKS_CNI_Policy est attachée au rôle, nous vous recommandons de la supprimer et de l'attacher à un rôle IAM qui est plutôt mappé au compte de service `aws-node` Kubernetes. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

6. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la relation d'approbation).
7. Vérifiez que la relation d'approbation contient la politique suivante. Si la relation d'approbation correspond à la politique ci-dessous, sélectionnez Annuler. Si la relation d'approbation ne correspond pas, copiez la politique dans la fenêtre Edit trust policy (Modifier la politique d'approbation) et sélectionnez Update policy (Mettre à jour la politique).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Création du rôle IAM de nœud Amazon EKS

Vous pouvez créer le rôle IAM du nœud avec le AWS Management Console ou le AWS CLI.

AWS Management Console

Pour créer votre rôle de nœud Amazon EKS dans la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Sur la page Rôles, choisissez Créer un rôle.
4. Sur la page Select trusted entity (Sélectionner une entité de confiance), procédez comme suit :
   a. Sous la section Type d'entité de confiance, sélectionnez Service AWS .
   b. Sous Cas d'utilisation, choisissez EC2.
   c. Choisissez Suivant.
5. Sur la page Ajouter des autorisations, associez une stratégie personnalisée ou procédez comme suit :
   a. Dans la zone Filter policies (Politiques de filtre), saisissez **`AmazonEKSWorkerNodePolicy`**.
   b. Cochez la case située à gauche d'WorkerNodePolicyAmazoneks dans les résultats de recherche.
   c. Sélectionnez Clear filters (Effacer les filtres).
   d. Dans la zone Filter policies (Politiques de filtre), saisissez **`AmazonEC2ContainerRegistryReadOnly`**.
   e. Cochez la case située à gauche d'AmazonEC2 ContainerRegistryReadOnly dans les résultats de recherche.

      La politique gérée AmazonEKS_CNI_Policy, ou une politique IPv6 que vous créez doit être également attachée à ce rôle ou à un autre rôle qui est mappé au compte de service `aws-node` Kubernetes. Nous vous recommandons d'attribuer la politique au rôle associé au compte de service Kubernetes au lieu de l'affecter à ce rôle. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).
   f. Choisissez Suivant.
6. Sur la page Name, review, and create (Nommer, vérifier et créer), procédez comme suit :

a. Pour Role name (Nom de rôle), saisissez un nom unique pour votre rôle, par exemple, **`AmazonEKSNodeRole`**.

b. Pour Description, remplacez le texte actuel par un texte descriptif tel que **`Amazon EKS - Node role`**.

c. Sous Ajouter des balises (Facultatif), ajoutez des métadonnées au rôle en attachant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.

d. Sélectionnez Créer un rôle.

AWS CLI

1. Exécutez la commande ci-dessous pour créer un fichier`node-role-trust-relationship.json`.

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

2. Créez le rôle IAM.

```
aws iam create-role \
  --role-name AmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-relationship.json"
```

3. Attachez deux politiques gérées IAM requises au rôle IAM.

```
aws iam attach-role-policy \
```

```
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name AmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name AmazonEKSNodeRole
```

4. Attachez l'une des politiques IAM suivantes au rôle IAM en fonction de la famille IP avec laquelle vous avez créé votre cluster. La politique doit être attachée à ce rôle ou à un rôle associé au compte de service Kubernetes `aws-node` utilisé pour le Amazon VPC CNI plugin for Kubernetes. Nous vous recommandons d'attribuer la politique au rôle associé au compte de service Kubernetes. Pour attribuer la politique au rôle associé au compte de service Kubernetes, veuillez consulter Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSNodeRole
     ```

   - IPv6

     1. Copiez la politique suivante et enregistrez-la dans un fichier appelé *vpc-cni-ipv6-policy*.json.

        ```
        {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": [
                        "ec2:AssignIpv6Addresses",
                        "ec2:DescribeInstances",
                        "ec2:DescribeTags",
                        "ec2:DescribeNetworkInterfaces",
                        "ec2:DescribeInstanceTypes"
                    ],
                    "Resource": "*"
                },
                {
                    "Effect": "Allow",
                    "Action": [
                        "ec2:CreateTags"
        ```

```
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. Créez la politique IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

3. Attachez la politique IAM au rôle IAM. Remplacez *111122223333* par votre ID de compte.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSNodeRole
```

Rôle IAM d'exécution de Pod Amazon EKS

Le rôle Pod d'exécution Amazon EKS est requis pour fonctionner Pods sur AWS Fargate l'infrastructure.

Lorsque votre cluster crée une Pods AWS Fargate infrastructure, les composants exécutés sur l'infrastructure Fargate doivent appeler les API en votre AWS nom. Cela leur permet d'effectuer des actions telles que l'extraction d'images de conteneurs depuis Amazon ECR ou le routage des journaux vers d'autres AWS services. Pour ce faire, le rôle d'exécution de Pod Amazon EKS fournit les autorisations IAM.

Lorsque vous créez un profil Fargate, vous devez spécifier un rôle d'exécution de Pod pour les composants Amazon EKS qui s'exécutent sur l'infrastructure Fargate utilisant le profil. Ce rôle est ajouté au contrôle d'accès basé sur les rôles (RBAC) du cluster Kubernetes à des fins d'autorisation. Cela permet au `kubelet` s'exécutant sur l'infrastructure Fargate de s'enregistrer dans votre cluster Amazon EKS, afin qu'il puisse apparaître dans votre cluster en tant que nœud.

> **Note**
>
> Le profil Fargate doit avoir un rôle IAM différent des groupes de nœuds Amazon EC2.

> **Important**
>
> Les conteneurs s'exécutant dans le Pod Fargate ne peuvent pas assumer les autorisations IAM associées à un rôle d'exécution de Pod. Pour autoriser les conteneurs de votre Pod Fargate à accéder à d' AWS autres services, vous devez utiliser. Rôles IAM pour les comptes de service

Avant de créer un profil Fargate, vous devez créer un rôle IAM avec l'`AmazonEKSFargatePodExecutionRolePolicy`.

Recherche d'un rôle d'exécution de Pod existant correctement configuré

Vous pouvez utiliser la procédure suivante pour vérifier et voir si votre compte possède déjà un rôle d'exécution de Pod Amazon EKS correctement configuré. Pour éviter le problème de sécurité du député confus, il est important que le rôle restreigne l'accès en fonction de `SourceArn`. Vous pouvez modifier le rôle d'exécution si nécessaire pour inclure la prise en charge des profils Fargate sur d'autres clusters.

Pour vérifier un rôle d'exécution de Pod Amazon EKS dans la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Sur la page Rôles, recherchez la liste des rôles pour Amazoneks. FargatePodExecutionRole Si le rôle n'existe pas, consultez Création du rôle d'exécution de Pod Amazon EKS pour le créer. Si le rôle existe, sélectionnez-le.
4. Sur la page FargatePodExecutionRoleAmazoneks, procédez comme suit :
   a. Choisissez Autorisations.
   b. Assurez-vous que la politique gérée par Amazon FargatePodExecutionRolePolicy Amazoneks est attachée au rôle.
   c. Choisissez Trust Relationships (Relations d'approbation).

d. Choisissez Edit trust policy (Modifier la politique).

5. Dans la page Edit trust policy (Modifier la politique), vérifiez que la relation d'approbation contient la politique suivante, ainsi qu'une ligne pour les profils Fargate sur votre cluster. Dans ce cas, sélectionnez Cancel (Annuler).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Si la politique correspond, mais ne possède pas de ligne spécifiant les profils Fargate sur votre cluster, vous pouvez ajouter la ligne suivante en haut de l'objet `ArnLike`. Remplacez *region-code* par la Région AWS dans laquelle se trouve votre cluster, *111122223333* par l'ID de votre compte et *my-cluster* par le nom de votre cluster.

```
"aws:SourceArn": "arn:aws:eks:region-code:111122223333:fargateprofile/my-cluster/
*",
```

Si la politique ne correspond pas, copiez entièrement la politique précédente dans le formulaire et choisissez Mettre à jour une politique. Remplacez *region-code* par la Région AWS dans laquelle se trouve votre cluster. Si vous souhaitez utiliser le même rôle Régions AWS dans tous les éléments de votre compte, remplacez le *code de région us-iso-east* par. * Remplacez *111122223333* par l'ID de votre compte et *my-cluster* par le nom de votre cluster. Si vous souhaitez utiliser le même rôle pour tous les clusters de votre compte, remplacez *my-cluster* par *.

Création du rôle d'exécution de Pod Amazon EKS

Si vous ne possédez pas encore le rôle Pod d'exécution Amazon EKS pour votre cluster, vous pouvez utiliser le AWS Management Console ou le AWS CLI pour le créer.

AWS Management Console

Pour créer un rôle d'exécution de PodAWS Fargate avec la AWS Management Console

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Sur la page Rôles, choisissez Créer un rôle.
4. Sur la page Select trusted entity (Sélectionner une entité de confiance), procédez comme suit :
   a. Sous la section Type d'entité de confiance, sélectionnez Service AWS .
   b. Dans la liste déroulante Use cases for other Services AWS (Cas d'utilisation pour d'autres), sélectionnez EKS.
   c. Sélectionnez EKS : Pod Fargate.
   d. Choisissez Suivant.
5. Sur la page Add permissions (Ajouter des autorisations), sélectionnez Next (Suivant).
6. Sur la page Name, review, and create (Nommer, vérifier et créer), procédez comme suit :
   a. Pour Role name (Nom de rôle), saisissez un nom unique pour votre rôle, par exemple, **`AmazonEKSFargatePodExecutionRole`**.
   b. Sous Ajouter des balises (Facultatif), ajoutez des métadonnées au rôle en attachant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.
   c. Sélectionnez Créer un rôle.
7. Sur la page Rôles, recherchez la liste des rôles pour Amazoneks. FargatePodExecutionRole Choisissez le rôle.
8. Sur la page FargatePodExecutionRoleAmazoneks, procédez comme suit :
   a. Choisissez Trust Relationships (Relations d'approbation).
   b. Choisissez Edit trust policy (Modifier la politique).

9. Dans la page Edit trust policy (Modifier la politique), procédez comme suit :

   a. Copiez le contenu suivant et collez-le dans le formulaire Edit trust policy (Modifier la politique). Remplacez le *code de région us-iso-east* par Région AWS celui dans lequel se trouve votre cluster. Si vous souhaitez utiliser le même rôle Régions AWS dans tous les éléments de votre compte, remplacez le *code de région us-iso-east* par. * Remplacez *111122223333* par l'ID de votre compte et *my-cluster* par le nom de votre cluster. Si vous souhaitez utiliser le même rôle pour tous les clusters de votre compte, remplacez *my-cluster* par *.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

   b. Choisissez Mettre à jour une politique.

AWS CLI

Pour créer un rôle AWS FargatePod d'exécution avec AWS CLI

1. Copiez le contenu suivant et collez-le dans un fichier nommé *pod-execution-role-trust-policy*.json. Remplacez le *code de région us-iso-east* par Région AWS celui dans lequel se trouve votre cluster. Si vous souhaitez utiliser le même rôle Régions AWS dans tous les éléments de votre compte, remplacez le *code de région us-iso-east* par. * Remplacez *111122223333* par l'ID de votre compte et *my-cluster* par le

nom de votre cluster. Si vous souhaitez utiliser le même rôle pour tous les clusters de votre compte, remplacez *my-cluster* par *.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Créez un rôle IAM d'exécution de Pod.

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-policy.json"
```

3. Attachez la politique IAM gérée par Amazon EKS au rôle.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Rôle IAM du connecteur Amazon EKS

Vous pouvez connecter Kubernetes des clusters pour les afficher dans votre AWS Management Console. Pour vous connecter à un cluster Kubernetes, créez un rôle IAM.

Rechercher un rôle de connecteur EKS existant

Vous pouvez utiliser la procédure suivante pour vérifier si votre compte possède déjà le rôle de connecteur Amazon EKS.

Pour vérifier le rôle **`AmazonEKSConnectorAgentRole`** dans la console IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Rôles.
3. Recherchez dans la liste des rôles `AmazonEKSConnectorAgentRole`. Si un rôle qui inclut `AmazonEKSConnectorAgentRole` n'existe pas, consultez Création du rôle de l'agent Amazon EKS Connector pour créer le rôle. Si un rôle qui inclut `AmazonEKSConnectorAgentRole` existe, sélectionnez le rôle pour afficher les politiques attachées.
4. Choisissez Permissions (Autorisations).
5. Assurez-vous que la politique gérée ConnectorAgentPolicy par Amazoneks est attachée au rôle. Si la politique est attachée, votre rôle de connecteur Amazon EKS est configuré correctement.
6. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la relation d'approbation).
7. Vérifiez que la relation d'approbation contient la politique suivante. Si la relation d'approbation correspond à la politique ci-dessous, sélectionnez Annuler. Si la relation d'approbation ne correspond pas, copiez la politique dans la fenêtre Edit trust policy (Modifier la politique d'approbation) et sélectionnez Update policy (Mettre à jour la politique).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Création du rôle de l'agent Amazon EKS Connector

Vous pouvez utiliser le AWS Management Console ou AWS CloudFormation pour créer le rôle d'agent du connecteur.

AWS CLI

1. Créez le fichier nommé `eks-connector-agent-trust-policy.json` qui contient le JSON suivant à utiliser pour le rôle IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Créez le fichier nommé `eks-connector-agent-policy.json` qui contient le JSON suivant à utiliser pour le rôle IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SsmControlChannel",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel"
            ],
            "Resource": "arn:aws:eks:*:*:cluster/*"
        },
        {
            "Sid": "ssmDataplaneOperations",
            "Effect": "Allow",
            "Action": [
```

```
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenDataChannel",
                "ssmmessages:OpenControlChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

3. Créez le rôle d'agent Amazon EKS Connector à l'aide de la politique d'approbation et de la politique que vous avez créées dans les listes précédentes.

```
aws iam create-role \
     --role-name AmazonEKSConnectorAgentRole \
     --assume-role-policy-document file://eks-connector-agent-trust-policy.json
```

4. Attachez la politique à votre rôle d'agent Amazon EKS Connector.

```
aws iam put-role-policy \
     --role-name AmazonEKSConnectorAgentRole \
     --policy-name AmazonEKSConnectorAgentPolicy \
     --policy-document file://eks-connector-agent-policy.json
```

AWS CloudFormation

Pour créer votre rôle d'agent du connecteur Amazon EKS avec AWS CloudFormation.

1. Enregistrez le AWS CloudFormation modèle suivant dans un fichier texte sur votre système local.

> **Note**
>
> Ce modèle crée également le rôle lié à un service qui serait autrement créé lorsque l'API `registerCluster` est appelée. Consultez Utilisation de rôles pour connecter un cluster Kubernetes à Amazon EKS pour plus de détails.

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Provisions necessary resources needed to register clusters in EKS'
```

```
Parameters: {}
Resources:
  EKSConnectorSLR:
    Type: AWS::IAM::ServiceLinkedRole
    Properties:
      AWSServiceName: eks-connector.amazonaws.com

  EKSConnectorAgentRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: [ 'sts:AssumeRole' ]
            Principal:
              Service: 'ssm.amazonaws.com'

  EKSConnectorAgentPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: EKSConnectorAgentPolicy
      Roles:
        - {Ref: 'EKSConnectorAgentRole'}
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateControlChannel' ]
            Resource:
            - Fn::Sub: 'arn:${AWS::Partition}:eks:*:*:cluster/*'
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateDataChannel',
 'ssmmessages:OpenDataChannel', 'ssmmessages:OpenControlChannel' ]
            Resource: "*"
Outputs:
  EKSConnectorAgentRoleArn:
    Description: The agent role that EKS connector uses to communicate with
 Services AWS.
    Value: !GetAtt EKSConnectorAgentRole.Arn
```

2. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.

3. Choisissez Create stack (Créer une pile) (soit avec de nouvelles ressources, soit avec des ressources existantes.
4. Pour Specify template (Spécifier un modèle), sélectionnez Upload a template file (Télécharger un fichier de modèle), puis choisissez Choose file (Choisir un fichier).
5. Choisissez le fichier que vous avez créé précédemment, puis choisissez Next (Suivant).
6. Pour Stack name (Nom de la pile), saisissez un nom pour votre rôle, par exemple `eksConnectorAgentRole`, puis choisissez Next (Suivant).
7. Sur la page Configurer les options de pile, choisissez Next (Suivant).
8. Sur la page Vérification, vérifiez vos informations, reconnaissez que la pile peut créer des ressources IAM, puis choisissez Create stack (Créer la pile).

AWS politiques gérées pour Amazon Elastic Kubernetes Service

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle politique Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez la section Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

AWS politique gérée : Amazoneks_CNI_Policy

Vous pouvez attacher `AmazonEKS_CNI_Policy` à vos entités IAM. Avant de créer un groupe de nœuds Amazon EC2, cette politique doit être attachée au rôle IAM de nœud ou à un rôle IAM qui est spécifiquement utilisé par le plugin Amazon VPC CNI plugin for Kubernetes. Ceci lui permet d'effectuer des actions en votre nom. Nous vous recommandons d'attacher la politique à un rôle qui n'est utilisé que par le plugin. Pour plus d'informations, consultez Utilisation du module complémentaire Amazon VPC CNI plugin for Kubernetes Amazon EKS et Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- **`ec2:*NetworkInterface`**et **`ec2:*PrivateIpAddresses`** — Permet au plugin Amazon VPC CNI d'effectuer des actions telles que le provisionnement d'interfaces réseau élastiques et d'adresses IP afin de fournir un réseau Pods aux applications exécutées dans Amazon EKS.
- **`ec2`**actions de lecture — Permet au plugin Amazon VPC CNI d'effectuer des actions telles que décrire des instances et des sous-réseaux pour voir le nombre d'adresses IP gratuites dans vos sous-réseaux Amazon VPC. Le VPC CNI peut utiliser les adresses IP libres de chaque sous-réseau pour sélectionner les sous-réseaux avec le plus grand nombre d'adresses IP libres à utiliser lors de la création d'une interface elastic network.

Pour consulter la dernière version du document de politique JSON, consultez Amazoneks_CNI_Policy dans le Managed Policy Reference Guide. AWS

AWS politique gérée : Amazoneks ClusterPolicy

Vous pouvez attacher `AmazonEKSClusterPolicy` à vos entités IAM. Avant de créer un cluster, vous devez disposer d'un rôle IAM de cluster auquel cette politique est attachée. Kubernetesles clusters gérés par Amazon EKS appellent d'autres AWS services en votre nom. Ils le font pour gérer les ressources que vous utilisez avec le service.

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- **`autoscaling`** : lit et met à jour la configuration d'un groupe Auto Scaling. Ces autorisations ne sont pas utilisées par Amazon EKS, mais restent dans la politique de compatibilité descendante.

- **`ec2`** : fonctionne avec des volumes et des ressources réseau associés aux nœuds Amazon EC2. Ceci est nécessaire pour que le plan de contrôle Kubernetes puisse joindre des instances à un cluster et allouer et gérer dynamiquement les volumes Amazon EBS demandés par des volumes persistants Kubernetes.
- **`elasticloadbalancing`** : fonctionne avec Elastic Load Balancers et ajoute des nœuds en tant que cibles. Ceci est nécessaire pour que le plan de contrôle Kubernetes puisse allouer de manière dynamique les équilibreurs de charge Elastic demandés par les services Kubernetes.
- **`iam`** : créez un rôle lié à un service Ceci est nécessaire pour que le plan de contrôle Kubernetes puisse allouer de manière dynamique les équilibreurs de charge Elastic demandés par les services Kubernetes.
- **`kms`** : lit une clé de AWS KMS. Cette condition est nécessaire pour que le plan de contrôle de Kubernetes prenne en charge le chiffrement des secrets de Kubernetes stockés dans `etcd`.

Pour consulter la dernière version du document de politique JSON, consultez ClusterPolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : Amazoneks FargatePodExecutionRolePolicy

Vous pouvez attacher `AmazonEKSFargatePodExecutionRolePolicy` à vos entités IAM. Avant de pouvoir créer un profil Fargate, vous devez créer un rôle d'exécution de Pod Fargate et y attacher cette politique. Pour plus d'informations, consultez Création d'un rôle d'exécution de Pod Fargate et AWS Fargate profil.

Cette politique accorde au rôle les autorisations permettant d'accéder aux autres ressources de AWS service requises pour exécuter Amazon EKS Pods sur Fargate.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- **`ecr`** : permet aux pods fonctionnant sur Fargate d'extraire des images de conteneur stockées dans Amazon ECR.

Pour consulter la dernière version du document de politique JSON, consultez FargatePodExecutionRolePolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : Amazoneks ForFargateServiceRolePolicy

Vous ne pouvez pas attacher `AmazonEKSForFargateServiceRolePolicy` à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à Amazon EKS d'effectuer des actions en votre nom. Pour plus d'informations, consultez AWSServiceRoleforAmazonEKSForFargate.

Cette politique accorde les autorisations nécessaires à Amazon EKS pour exécuter des tâches Fargate. La politique n'est utilisée que si vous avez des nœuds Fargate.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes.

- **`ec2`** : créez et supprimez des interfaces réseau Elastic et décrivez les interfaces et les ressources réseau Elastic. Ceci est nécessaire pour que le service Amazon EKS Fargate puisse configurer les réseaux VPC requis pour les pods Fargate.

Pour consulter la dernière version du document de politique JSON, consultez ForFargateServiceRolePolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : Amazoneks ServicePolicy

Vous pouvez attacher `AmazonEKSServicePolicy` à vos entités IAM. Les clusters créés avant le 16 avril 2020 nécessitaient la création d'un rôle IAM et l'ajout de cette politique. Les clusters créés à partir du 16 avril 2020 ne nécessitent pas la création d'un rôle ni l'affectation de cette politique. Lorsque vous créez un cluster à l'aide d'un principal IAM `iam:CreateServiceLinkedRole` autorisé, le rôle lié au service AWS ServiceRoleforAmazonEKS est automatiquement créé pour vous. AWS politique gérée : Amazoneks ServiceRolePolicy est attaché au rôle lié au service.

Cette politique permet à Amazon EKS de créer et de gérer les ressources nécessaires à l'exploitation des clusters Amazon EKS.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes.

- **`eks`** : mettez à jour la version Kubernetes de votre cluster après avoir démarré une mise à jour. Cette autorisation n'est pas utilisée par Amazon EKS, mais reste dans la politique de compatibilité descendante.
- **`ec2`** : travaillez avec les interfaces réseau Elastic et d'autres ressources réseau et identifications. Cela est requis par Amazon EKS pour configurer la mise en réseau qui facilite la communication entre les nœuds et le plan de contrôle Kubernetes.
- **`route53`** : associez un VPC à une zone hébergée. Cela est requis par Amazon EKS pour activer la mise en réseau de points de terminaison privés pour votre serveur d'API de cluster Kubernetes.
- **`logs`** : journalisez les événements. Cela est nécessaire pour qu'Amazon EKS puisse expédier les journaux du plan de Kubernetes contrôle à CloudWatch.
- **`iam`** : créez un rôle lié à un service Ceci est nécessaire pour qu'Amazon EKS puisse créer le rôle lié à un service `AWSServiceRoleForAmazonEKS` en votre nom.

Pour consulter la dernière version du document de politique JSON, consultez ServicePolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : Amazoneks ServiceRolePolicy

Vous ne pouvez pas attacher `AmazonEKSServiceRolePolicy` à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à Amazon EKS d'effectuer des actions en votre nom. Pour plus d'informations, consultez Autorisations du rôle lié à un service pour Amazon EKS. Lorsque vous créez un cluster à l'aide d'un principal IAM `iam:CreateServiceLinkedRole` autorisé, le rôle lié au service AWS ServiceRoleforAmazonEKS est automatiquement créé pour vous et cette politique y est associée.

Cette politique permet au rôle lié au service d'appeler les AWS services en votre nom.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes.

- **`ec2`** : créez et décrivez les interfaces réseau Elastic et les instances Amazon EC2, le groupe de sécurité du cluster et le VPC requis pour créer un cluster.
- **`iam`** : répertoriez toutes les politiques gérées associées à un rôle IAM. Cette condition est nécessaire pour permettre à Amazon EKS de répertorier et valider toutes les politiques et autorisations gérées requises pour créer un cluster.

- Associer un VPC à une zone hébergée : cela est requis par Amazon EKS pour activer la mise en réseau de points de terminaison privés pour votre serveur d'API de cluster Kubernetes.
- Enregistrer les événements : cela est nécessaire pour qu'Amazon EKS puisse expédier les journaux du plan de Kubernetes contrôle à CloudWatch.

Pour consulter la dernière version du document de politique JSON, consultez ServiceRolePolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : AmazoneKSVPC ResourceController

Vous pouvez associer la politique `AmazonEKSVPCResourceController` à vos identités IAM. Si vous utilisez des groupes de sécurité pour les Pods, vous devez attacher cette politique à votre Rôle IAM de cluster Amazon EKS pour effectuer des actions en votre nom.

Cette politique accorde les autorisations de rôle de cluster pour gérer les interfaces réseau Elastic et les adresses IP pour les nœuds.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- **`ec2`** : gérez les interfaces réseau Elastic et les adresses IP pour prendre en charge les groupes de sécurité de Pod et les nœuds Windows.

Pour consulter la dernière version du document de politique JSON, consultez AmazoneKSVPC ResourceController dans le AWS Managed Policy Reference Guide.

AWS politique gérée : Amazoneks WorkerNodePolicy

Vous pouvez attacher `AmazonEKSWorkerNodePolicy` à vos entités IAM. Vous devez attacher cette politique à un rôle IAM de nœud spécifié lorsque vous créez des nœuds Amazon EC2 qui permettent à Amazon EKS d'effectuer des actions en votre nom. Si vous créez un groupe de nœuds à l'aide de `eksctl`, il crée le rôle IAM de nœud et attache automatiquement cette politique au rôle.

Cette politique accorde aux nœuds Amazon EKS Amazon EC2 les autorisations de connexion aux clusters Amazon EKS.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- **`ec2`** : lit les informations sur le volume de l'instance et le réseau. Ceci est nécessaire pour que les nœuds Kubernetes puissent donner des informations sur les ressources Amazon EC2 requises afin que le nœud puisse rejoindre le cluster Amazon EKS.
- **`eks`** : décrivez éventuellement le cluster dans le cadre de l'amorçage des nœuds.
- **`eks-auth:AssumeRoleForPodIdentity`** : permet de récupérer les informations d'identification pour les charges de travail EKS sur le nœud. Cela est nécessaire pour que l'identité du pod EKS fonctionne correctement.

Pour consulter la dernière version du document de politique JSON, consultez WorkerNodePolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : AWSServiceRoleForAmazonEKSNodegroup

Vous ne pouvez pas attacher `AWS ServiceRoleForAmazonEKSNodegroup` à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à Amazon EKS d'effectuer des actions en votre nom. Pour plus d'informations, consultez Autorisations du rôle lié à un service pour Amazon EKS.

Cette politique accorde au rôle les autorisations `AWS ServiceRoleForAmazonEKSNodegroup` qui lui permettent de créer et de gérer les groupes de nœuds Amazon EC2 dans votre compte.

Détails de l'autorisation

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- **`ec2`** : fonctionne avec des groupes de sécurité, des identifications et des modèles de lancement. Ceci est requis pour les groupes de nœuds gérés Amazon EKS pour activer la configuration de l'accès à distance. En outre, les groupes de nœuds gérés par Amazon EKS créent un modèle de lancement en votre nom. Cela permet de configurer le groupe Amazon EC2 Auto Scaling qui prend en charge chaque groupe de nœuds gérés.
- **`iam`** : crée un rôle lié à un service et passe un rôle. Ceci est requis par les groupes de nœuds gérés Amazon EKS pour gérer les profils d'instance pour le rôle transmis lors de la création d'un groupe de nœuds gérés. Ce profil d'instance est utilisé par les instances Amazon EC2 lancées dans le cadre d'un groupe de nœuds gérés. Amazon EKS doit créer des rôles liés au service pour

d'autres services tels que les groupes Amazon EC2 Auto Scaling. Ces autorisations sont utilisées dans la création d'un groupe de nœuds gérés.

- **`autoscaling`** : utilisation des groupes Auto Scaling de sécurité. Ceci est requis par les groupes de nœuds gérés par Amazon EKS pour gérer le groupe Amazon EC2 Auto Scaling qui sauvegarde chaque groupe de nœuds gérés. Il est également utilisé pour prendre en charge des fonctionnalités telles que l'expulsion des Pods lorsque les nœuds sont résiliés ou recyclés pendant les mises à jour des groupes de nœuds.

Pour consulter la dernière version du document de politique JSON, consultez AWSServiceRoleForAmazonEKSNodegrouple Guide de référence des politiques AWS gérées.

AWS politique gérée : AmazoneBSCSI DriverPolicy

La politique `AmazonEBSCSIDriverPolicy` permet au pilote Amazon EBS Container Storage Interface (CSI) de créer, de modifier, d'attacher, de détacher et de supprimer des volumes en votre nom. Il accorde également au pilote EBS CSI les autorisations nécessaires pour créer et supprimer des instantanés, ainsi que pour répertorier vos instances, volumes et instantanés.

Pour consulter la dernière version du document de politique JSON, consultez AmazoneBSCSI DriverServiceRolePolicy dans le AWS Managed Policy Reference Guide.

AWS politique gérée : AmazonefSCSI DriverPolicy

La politique `AmazonEFSCSIDriverPolicy` permet à l'interface CSI (Amazon EFS Container Storage Interface) de créer et de supprimer des points d'accès en votre nom. Il autorise également le pilote CSI Amazon EFS à répertorier vos points d'accès, vos systèmes de fichiers, vos cibles de montage et les zones de disponibilité Amazon EC2.

Pour consulter la dernière version du document de politique JSON, consultez AmazonEFSCSI DriverServiceRolePolicy dans le Guide de référence des politiques AWS gérées.

AWS politique gérée : Amazoneks LocalOutpostClusterPolicy

Vous ne pouvez pas attacher cette politique à des entités IAM. Avant de créer un cluster local, vous devez associer cette politique à votre rôle de cluster. Kubernetesles clusters gérés par Amazon EKS appellent d'autres AWS services en votre nom. Ils le font pour gérer les ressources que vous utilisez avec le service.

La politique `AmazonEKSLocalOutpostClusterPolicy` inclut les autorisations suivantes :

- **`ec2`** – Autorisations requises pour que les instances Amazon EC2 rejoignent correctement le cluster en tant qu'instances du plan de contrôle.
- **`ssm`** – Permet à Amazon EC2 Systems Manager de se connecter à l'instance du plan de contrôle, qui est utilisée par Amazon EKS pour communiquer et gérer le cluster local dans votre compte.
- **`logs`**— Permet aux instances de transmettre des journaux à Amazon CloudWatch.
- **`secretsmanager`**— Permet aux instances d'obtenir et de supprimer des données de démarrage pour les instances du plan de contrôle en toute sécurité. AWS Secrets Manager
- **`ecr`** – Permet aux Pods et aux conteneurs exécutés sur des instances du plan de contrôle d'extraire des images de conteneur stockées dans Amazon Elastic Container Registry.

Pour consulter la dernière version du document de politique JSON, consultez LocalOutpostClusterPolicyAmazoneks dans le Managed Policy Reference AWS Guide.

AWS politique gérée : Amazoneks LocalOutpostServiceRolePolicy

Vous ne pouvez pas attacher cette politique à vos entités IAM. Lorsque vous créez un cluster à l'aide d'un principal IAM disposant de l'autorisation `iam:CreateServiceLinkedRole`, Amazon EKS crée automatiquement le rôle lié au service `AWSServiceRoleforAmazonEKSLocalOutpost` pour vous et y attache cette politique. Cette politique permet au rôle lié au service d'appeler des AWS services en votre nom pour les clusters locaux.

La politique `AmazonEKSLocalOutpostServiceRolePolicy` inclut les autorisations suivantes :

- **`ec2`** – Permet à Amazon EKS de travailler avec la sécurité, le réseau et d'autres ressources pour pouvoir lancer et gérer les instances du plan de contrôle dans votre compte.
- **`ssm`** – Permet à Amazon EC2 Systems Manager de se connecter aux instances du plan de contrôle, qui sont utilisées par Amazon EKS pour communiquer et gérer le cluster local dans votre compte.
- **`iam`** – Permet à Amazon EKS de gérer le profil d'instance associé aux instances du plan de contrôle.
- **`secretsmanager`**— Permet à Amazon EKS de placer des données de démarrage pour les instances du plan de contrôle AWS Secrets Manager afin qu'elles puissent être référencées de manière sécurisée lors du démarrage des instances.
- **`outposts`** – Permet à Amazon EKS d'obtenir des informations sur Outpost à partir de votre compte afin de lancer avec succès un cluster local dans un Outpost.

Pour consulter la dernière version du document de politique JSON, consultez LocalOutpostServiceRolePolicyAmazoneks dans le Managed Policy Reference AWS Guide.

Amazon EKS met à jour les politiques AWS gérées

Consultez les informations relatives aux mises à jour des politiques AWS gérées pour Amazon EKS 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 dans la page de l'historique des documents Amazon EKS.

Modification	Description	Date
Amazoneks_CNI_Policy — Mise à jour d'une politique existante	Amazon EKS a ajouté de nouvelles `ec2:DescribeSubnets` autorisations pour vous permettre de voir le nombre d'adresses IP gratuites dans vos sous-réseaux Amazon VPC. Amazon VPC CNI plugin for Kubernetes Le VPC CNI peut utiliser les adresses IP libres de chaque sous-réseau pour sélectionner les sous-réseaux avec le plus grand nombre d'adresses IP libres à utiliser lors de la création d'une interface elastic network.	4 mars 2024
WorkerNodePolicyAmazoneks — Mise à jour d'une politique existante	Amazon EKS a ajouté de nouvelles autorisations pour permettre les identités du pod EKS. L'agent d'identité du pod Amazon EKS utilise le rôle de nœud.	26 novembre 2023
Présentation d'Amazon EFSCSI DriverPolicy.	AWS a présenté le`AmazonEFS CSIDriverPolicy` .	26 juillet 2023

Modification	Description	Date
Autorisations ajoutées à Amazoneks. ClusterPolicy	Ajout d'une autorisation `ec2:DescribeAvailabilityZones` permettant à Amazon EKS d'obtenir les détails des zones de disponibilité lors de la découverte automatique des sous-réseaux lors de la création d'équilibreurs de charge.	7 février 2023
Conditions de politique mises à jour dans AmazoneBSCSI DriverPolicy.	Conditions de stratégie non valides supprimées comportant des caractères génériques dans le champ clé `StringLike` . Nouvelle condition `ec2:ResourceTag/kubernetes.io/created-for/pvc/name: "*"` ajoutée à `ec2:DeleteVolume` , permettant au pilote EBS CSI de supprimer les volumes créés par le plug-in intégré à l'arborescence.	17 novembre 2022
Autorisations ajoutées à Amazoneks. LocalOutpostServiceRolePolicy	Ajout de `ec2:DescribeVPCAttribute` , `ec2:GetConsoleOutput` et `ec2:DescribeSecret` pour permettre une meilleure validation des prérequis et un meilleur contrôle du cycle de vie géré. Également, ajout de `ec2:DescribePlacementGroups` et `"arn:aws:ec2:*:*:placement-group/*"` à `ec2:RunInstances` pour permettre le contrôle du placement des instances Amazon EC2 du plan de contrôle sur les Outposts.	24 octobre 2022

Modification	Description	Date
Mettez à jour les autorisations d'Amazon Elastic Container Registry dans Amazoneks. LocalOutpostClusterPolicy	Déplacement de l'action `ecr:GetDownloadUrlForLayer` de toutes les sections de ressources vers une section délimitée. Ajout de la ressource `arn:aws:ecr:*:*:repository/eks/*` . Suppression de la ressource `arn:aws:ecr:*:*:repository/eks/eks-certificates-controller-public` . Cette ressource est couverte par la ressource `arn:aws:ecr:*:*:repository/eks/*` ajoutée.	20 octobre 2022
Autorisations ajoutées à Amazoneks. LocalOutpostClusterPolicy	Ajout du référentiel Amazon Elastic Container Registry `arn:aws:ecr:*:*:repository/kubelet-config-updater` pour que les instances du plan de contrôle du cluster puissent mettre à jour des arguments `kubelet`.	31 août 2022
Présentation d'Amazoneks. LocalOutpostClusterPolicy	AWS a présenté le`AmazonEKS LocalOutpostClusterPolicy` .	24 août 2022
Présentation d'Amazoneks. LocalOutpostServiceRolePolicy	AWS a présenté le`AmazonEKS LocalOutpostServiceRolePolicy` .	23 août 2022
Présentation d'Amazon BSCSI. DriverPolicy	AWS a présenté le`AmazonEBS CSIDriverPolicy` .	4 avril 2022

Modification	Description	Date
Autorisations ajoutées à Amazoneks. WorkerNodePolicy	Ajout de `ec2:DescribeInstanceTypes` pour activer les AMI optimisées pour Amazon EKS qui peuvent détecter automatiquement les propriétés au niveau de l'instance.	21 mars 2022
Autorisations ajoutées à AWSServiceRoleForAmazonEKSNodegroup.	Ajout d'autorisations `autoscaling:EnableMetricsCollection` permettant à Amazon EKS d'activer la collecte de métriques.	13 décembre 2021
Autorisations ajoutées à Amazoneks. ClusterPolicy	Ajout de `ec2:DescribeAccountAttributes` , `ec2:DescribeAddresses` et `ec2:DescribeInternetGateways` pour autoriser Amazon EKS à créer un rôle lié à un service pour un Network Load Balancer.	17 juin 2021
Amazon EKS a commencé à assurer le suivi des modifications.	Amazon EKS a commencé à suivre les modifications apportées AWS à ses politiques gérées.	17 juin 2021

Dépannage IAM

Cette rubrique traite de certaines erreurs courantes que vous pouvez rencontrer lorsque vous utilisez Amazon EKS avec IAM, ainsi que des solutions.

AccessDeniedException

Si vous recevez un message `AccessDeniedException` lors de l'appel d'une opération d' AWS API, cela signifie que les informations d'identification principales IAM que vous utilisez ne disposent pas des autorisations requises pour effectuer cet appel.

```
An error occurred (AccessDeniedException) when calling the DescribeCluster operation:
User: arn:aws:iam::111122223333:user/user_name is not authorized to perform:
```

```
eks:DescribeCluster on resource: arn:aws:eks:region:111122223333:cluster/my-cluster
```

Dans l'exemple de message ci-précédent, l'utilisateur n'est pas autorisé à appeler l'opération d'API `DescribeCluster` Amazon EKS. Pour accorder des autorisations d'administrateur Amazon EKS à un principal IAM, consultez la rubrique Exemples de politiques basées sur l'identité d'Amazon EKS.

Pour plus d'informations générales sur IAM, consultez Contrôle de l'accès à l'aide de politiques dans le Guide de l'utilisateur IAM.

Vous ne voyez pas Nodes (Nœuds) sur l'onglet Compute (Calcul) ou quoi que ce soit sur l'onglet Resources (Ressources) et vous recevez une erreur dans la AWS Management Console.

Vous pouvez voir un message d'erreur de la console indiquant `Your current user or role does not have access to Kubernetes objects on this EKS cluster`. Assurez-vous que l'utilisateur principal IAM AWS Management Console avec lequel vous utilisez dispose des autorisations nécessaires. Pour plus d'informations, consultez Autorisations nécessaires.

aws-auth **`ConfigMap`** n'accorde pas l'accès au cluster

L'authentification AWS IAM n'autorise pas de chemin dans l'ARN de rôle utilisé dans la `ConfigMap`. Par conséquent, avant de spécifier `rolearn`, supprimez le chemin d'accès. Remplacez, par exemple, `arn:aws:iam::111122223333:role/team/developers/eks-admin` par `arn:aws:iam::111122223333:role/eks-admin`.

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 politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon EKS.

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 EKS. Toutefois, l'action nécessite que le service ait des autorisations accordées par une fonction du 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 stratégies 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 mon AWS compte à accéder à mes ressources Amazon EKS

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 EKS est compatible avec ces fonctionnalités, consultez Fonctionnement d'Amazon EKS 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 découvrir quelle est la différence entre l'utilisation des rôles et l'utilisation des politiques basées sur les ressources pour l'accès intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.

Les conteneurs Pod reçoivent l'erreur suivante : **`An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`**

Vos conteneurs reçoivent cette erreur si votre application envoie explicitement des demandes au point de terminaison AWS STS global (`https://sts.amazonaws`) et si votre compte de Kubernetes service est configuré pour utiliser un point de terminaison régional. Vous pouvez résoudre le problème avec l'une des options suivantes :

- Mettez à jour le code de votre application pour supprimer les appels explicites au point de terminaison AWS STS global.
- Mettez à jour le code de votre application pour effectuer des appels explicites vers des points de terminaison régionaux tels que `https://sts.us-west-2.amazonaws.com`. Votre application doit avoir une redondance intégrée pour en choisir une autre Région AWS en cas de panne du service dans la Région AWS. Pour plus d'informations, consultez Gestion de AWS STS dans une Région AWS dans le Guide de l'utilisateur IAM.
- Configurez vos comptes de service pour utiliser le point de terminaison mondial. Toutes les versions antérieures à la version `1.22` utilisaient par défaut le point de terminaison global, mais la version `1.22` et les clusters ultérieurs utilisent par défaut le point de terminaison régional. Pour plus d'informations, voir Configuration du AWS Security Token Service point de terminaison pour un compte de service.

Rôles et utilisateurs Kubernetes par défaut créés par Amazon EKS

Lorsque vous créez un cluster Kubernetes, plusieurs identités Kubernetes par défaut sont créées sur ce cluster pour le bon fonctionnement de Kubernetes. Amazon EKS crée des identités Kubernetes pour chacun de ses composants par défaut. Les identités fournissent un contrôle d'autorisation basé sur les rôles (RBAC) Kubernetes pour les composants du cluster. Pour plus d'informations, consultez Utilisation de l'autorisation RBAC dans la documentation Kubernetes.

Lorsque vous installez des modules complémentaires facultatifs sur votre cluster, des identités Kubernetes supplémentaires peuvent être ajoutées à votre cluster. Pour plus d'informations sur les identités non abordées dans cette rubrique, consultez la documentation du module complémentaire.

Vous pouvez consulter la liste des identités Kubernetes créées par Amazon EKS sur votre cluster à l'aide de la AWS Management Console ou de l'outil de ligne de commande `kubectl`. Toutes les

identités des utilisateurs apparaissent dans les journaux `kube` d'audit mis à votre disposition via Amazon CloudWatch.

AWS Management Console

Prérequis

Le principal IAM que vous utilisez doit disposer des autorisations décrites dans la rubrique Autorisations nécessaires.

Pour consulter les identités créées par Amazon EKS à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans la liste Clusters, sélectionnez le cluster qui contient les identités que vous souhaitez afficher.
3. Sélectionnez l'onglet Ressources.
4. Sous Resource types (Types de ressources), sélectionnez Authorization (Autorisation).
5. Choisissez, ClusterRolesClusterRoleBindings, Rôles ou RoleBindings. Toutes les ressources précédées de eks sont créées par Amazon EKS. Les ressources d'identité supplémentaires créées par Amazon EKS sont les suivantes :

   - Le ClusterRoleet ClusterRoleBindingnommé aws-node. Les ressources aws-node prennent en charge le Amazon VPC CNI plugin for Kubernetes, qu'Amazon EKS installe sur tous les clusters.
   - Un ClusterRolenom vpc-resource-controller-roleet un ClusterRoleBindingnom vpc-resource-controller-rolebinding. Ces ressources prennent en charge le contrôleur de ressources Amazon VPC, qu'Amazon EKS installe sur tous les clusters.

   Outre les ressources visibles sur la console, les identités d'utilisateur spéciales suivantes sont disponibles sur votre cluster, même si elles n'apparaissent pas dans la configuration du cluster :

   - **`eks:cluster-bootstrap`** : utilisée pour les opérations `kubectl` lors de l'amorçage du cluster.
   - **`eks:support-engineer`** : utilisée pour les opérations de gestion des clusters.

6. Choisissez une ressource spécifique pour afficher les détails la concernant. Par défaut, les informations s'affichent en mode Structured view (Vue structurée). Dans le coin supérieur droit de la page de détails, vous pouvez sélectionner Raw view (Vue brute) pour afficher toutes les informations relative à la ressource.

Kubectl

Prérequis

L'entité que vous utilisez (AWS Identity and Access Management (IAM) ou OpenID Connect ()OIDC) pour répertorier les Kubernetes ressources du cluster doit être authentifiée par IAM ou par votre OIDC fournisseur d'identité. L'entité doit être autorisée à utiliser les verbes Kubernetes `get` et `list` pour les ressources `Role`, `ClusterRole`, `RoleBinding` et `ClusterRoleBinding` de votre cluster avec lesquelles vous souhaitez que l'entité travaille. Pour plus d'informations sur la façon dont vous pouvez autoriser votre cluster à accéder aux entités IAM, consultez the section called “Autoriser l'accès aux API Kubernetes”. Pour plus d'informations sur la façon dont vous pouvez autoriser votre cluster à accéder aux entités authentifiées par votre propre fournisseur OIDC, consultez Authentifier les utilisateurs de votre cluster auprès d'un fournisseur d'OpenID Connectidentité.

Pour consulter les identités créées par Amazon EKS à l'aide de **`kubectl`**

Exécutez la commande correspondant au type de ressource que vous souhaitez consulter. Toutes les ressources renvoyées qui sont précédées de eks sont créées par Amazon EKS. Outre les ressources renvoyées dans la sortie des commandes, les identités d'utilisateur spéciales suivantes sont disponibles sur votre cluster, même si elles n'apparaissent pas dans la configuration du cluster :

- **`eks:cluster-bootstrap`** : utilisée pour les opérations `kubectl` lors de l'amorçage du cluster.
- **`eks:support-engineer`** : utilisée pour les opérations de gestion des clusters.

ClusterRoles— `ClusterRoles` sont limités à votre cluster, de sorte que toute autorisation accordée à un rôle s'applique aux ressources de n'importe quel espace de Kubernetes noms du cluster.

La commande suivante renvoie tous les `ClusterRoles` Kubernetes créés par Amazon EKS sur votre cluster.

```
kubectl get clusterroles | grep eks
```

Outre les `ClusterRoles` renvoyés dans la sortie, les `ClusterRoles` suivants sont disponibles.

- **`aws-node`** : ce `ClusterRole` prend en charge le Amazon VPC CNI plugin for Kubernetes, qu'Amazon EKS installe sur tous les clusters.
- **`vpc-resource-controller-role`** : ce `ClusterRole` prend en charge le contrôleur de ressources Amazon VPC, qu'Amazon EKS installe sur tous les clusters.

Pour afficher la spécification d'un `ClusterRole`, remplacez *`eks:k8s-metrics`* dans la commande suivante par un `ClusterRole` renvoyé dans la sortie de la commande précédente. L'exemple suivant renvoie la spécification du `ClusterRole` *`eks:k8s-metrics`*.

```
kubectl describe clusterrole eks:k8s-metrics
```

L'exemple qui suit illustre un résultat.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names  Verbs
  ---------         -----------------  --------------  -----
                    [/metrics]         []              [get]
  endpoints         []                 []              [list]
  nodes             []                 []              [list]
  pods              []                 []              [list]
  deployments.apps  []                 []              [list]
```

ClusterRoleBindings— `ClusterRoleBindings` sont limités à votre cluster.

La commande suivante renvoie tous les `ClusterRoleBindings` Kubernetes créés par Amazon EKS sur votre cluster.

```
kubectl get clusterrolebindings | grep eks
```

En plus des `ClusterRoleBindings` renvoyés dans la sortie, les `ClusterRoleBindings` suivants sont disponibles.

- **aws-node** : ce `ClusterRoleBinding` prend en charge le Amazon VPC CNI plugin for Kubernetes, qu'Amazon EKS installe sur tous les clusters.
- **vpc-resource-controller-rolebinding** : ce `ClusterRoleBinding` prend en charge le contrôleur de ressources Amazon VPC, qu'Amazon EKS installe sur tous les clusters.

Pour afficher la spécification d'un `ClusterRoleBinding`, remplacez *eks:k8s-metrics* dans la commande suivante par un `ClusterRoleBinding` renvoyé dans la sortie de la commande précédente. L'exemple suivant renvoie la spécification du `ClusterRoleBinding` *eks:k8s-metrics*.

```
kubectl describe clusterrolebinding eks:k8s-metrics
```

L'exemple qui suit illustre un résultat.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Roles : les `Roles` sont limités à un espace de noms Kubernetes. Tous les `Roles` créés par Amazon EKS sont limités à l'espace de noms `kube-system`.

La commande suivante renvoie tous les `Roles` Kubernetes créés par Amazon EKS sur votre cluster.

```
kubectl get roles -n kube-system | grep eks
```

Pour afficher la spécification d'un `Role`, remplacez *eks:k8s-metrics* dans la commande suivante par le nom d'un `Role` renvoyé dans la sortie de la commande précédente. L'exemple suivant renvoie la spécification du `Role` *eks:k8s-metrics*.

```
kubectl describe role eks:k8s-metrics -n kube-system
```

L'exemple qui suit illustre un résultat.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names             Verbs
  ---------         -----------------  --------------             -----
  daemonsets.apps   []                 [aws-node]                 [get]
  deployments.apps  []                 [vpc-resource-controller]  [get]
```

RoleBindings— `RoleBindings` sont limités à un espace de Kubernetes noms. Tous les `RoleBindings` créés par Amazon EKS sont limités à l'espace de noms `kube-system`.

La commande suivante renvoie tous les `RoleBindings` Kubernetes créés par Amazon EKS sur votre cluster.

```
kubectl get rolebindings -n kube-system | grep eks
```

Pour afficher la spécification d'un `RoleBinding`, remplacez *`eks:k8s-metrics`* dans la commande suivante par un `RoleBinding` renvoyé dans la sortie de la commande précédente. L'exemple suivant renvoie la spécification du `RoleBinding` *`eks:k8s-metrics`*.

```
kubectl describe rolebinding eks:k8s-metrics -n kube-system
```

L'exemple qui suit illustre un résultat.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  Role
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Validation de la conformité pour Amazon Elastic Kubernetes Service

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 EKS

L'infrastructure mondiale d'AWS est construite autour de zones de disponibilité et de Régions AWS. Les Régions AWSfournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par un réseau à latence faible, à débit élevé et à forte redondance. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone de disponibilité à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Amazon EKS exécute et met à l'échelle le plan de contrôle Kubernetes sur plusieurs zones de disponibilité AWS, ce qui permet d'assurer une haute disponibilité. Amazon EKS met automatiquement à l'échelle les instances du plan de contrôle en fonction de la charge, détecte et remplace les instances du plan de contrôle non saines, et applique automatiquement des correctifs au plan de contrôle. Après que vous ayez lancé une mise à jour de version, Amazon EKS met à jour votre plan de contrôle pour vous, en maintenant la haute disponibilité du plan de contrôle pendant la mise à jour.

Ce plan de contrôle se compose d'au moins deux instances de serveur API et de trois instances `etcd` qui s'exécutent sur trois zones de disponibilité au sein d'une Région AWS. Amazon EKS :

- Surveille activement la charge sur les instances de plan de contrôle et les met automatiquement à l'échelle pour garantir des performances élevées.
- Détecte et remplace automatiquement les instances de plan de contrôle défectueuses. Ces instances sont redémarrées dans les zones de disponibilité au sein de la Région AWS, selon le cas.
- Exploite l'architecture des Régions AWS afin de maintenir une haute disponibilité. De ce fait, Amazon EKS peut offrir un accord de niveau de service (SLA) pour la disponibilité des points de terminaison de serveur d'API.

Pour plus d'informations sur les Régions AWS et les zones de disponibilité, consultez Infrastructure mondiale d'AWS.

Sécurité de l'infrastructure dans Amazon EKS

En tant que service géré, Amazon Elastic Kubernetes Service est protégé par la sécurité du réseau mondial. AWS Pour plus d'informations sur les services AWS de sécurité et sur la manière dont AWS l'infrastructure est protégée, consultez la section Sécurité du AWS cloud. Pour concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à Amazon EKS via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Lorsque vous créez un cluster Amazon EKS, vous spécifiez les sous-réseaux VPC à utiliser par votre cluster. Amazon EKS nécessite des sous-réseaux dans au moins deux zones de disponibilité. Nous recommandons un VPC avec des sous-réseaux publics et privés afin que Kubernetes puisse créer des équilibreurs de charge publics dans les sous-réseaux publics qui équilibrent la charge du trafic vers des Pods s'exécutant sur des nœuds qui se trouvent dans des sous-réseaux privés.

Pour plus d'informations sur les considérations VPC, consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux.

Si vous créez votre VPC et vos groupes de nœuds à l'aide des AWS CloudFormation modèles fournis dans la Démarrer avec Amazon EKS procédure pas à pas, votre plan de contrôle et vos groupes de sécurité de nœuds sont configurés selon nos paramètres recommandés.

Pour plus d'informations sur les considérations des groupes de sécurité, consultez Considérations et exigences relatives aux groupes de sécurité Amazon EKS.

Lorsque vous créez un cluster, Amazon EKS crée un point de terminaison pour le serveur d'API Kubernetes géré que vous utilisez pour communiquer avec votre cluster (à l'aide d'outils de gestion Kubernetes tels que `kubectl`). Par défaut, ce point de terminaison du serveur d'API est public sur Internet, et l'accès au serveur d'API est sécurisé à l'aide d'une combinaison de AWS Identity and Access Management (IAM) et de contrôle d'accès basé sur les Kubernetes rôles (RBAC) natif.

Vous pouvez activer l'accès privé au serveur d'API Kubernetes pour que toutes les communications entre vos nœuds et le serveur d'API restent au sein de votre VPC. Vous pouvez limiter les adresses IP qui peuvent accéder à votre serveur API à partir d'Internet, ou désactiver complètement l'accès Internet au serveur d'API.

Pour plus d'informations sur la modification de l'accès au point de terminaison de cluster, consultez Modification de l'accès au point de terminaison de cluster.

Vous pouvez implémenter des stratégies réseau Kubernetes avec Amazon VPC CNI ou des outils tiers tels que Projet Calico. Pour plus d'informations sur l'utilisation de l'Amazon VPC CNI pour les stratégies réseau, consultez Configurez votre cluster pour les stratégies réseau Kubernetes. Project Calico est un projet open source tiers. Pour plus d'informations, consultez la documentation Project Calico.

Configuration et analyse des vulnérabilités dans Amazon EKS

La sécurité est une considération essentielle pour la configuration et la maintenance des clusters et des applications Kubernetes. Vous trouverez ci-dessous la liste des ressources vous permettant

d'analyser la configuration de sécurité de vos clusters EKS, des ressources vous permettant de détecter les vulnérabilités et des intégrations avec des AWS services capables de réaliser cette analyse à votre place.

Le test de référence du Center for Internet Security (CIS) pour Amazon EKS

Le test de Kubernetesréférence du Center for Internet Security (CIS) fournit des conseils pour les configurations de sécurité Amazon EKS. Le benchmark :

- S'applique aux nœuds Amazon EC2 (gérés et autogérés) dans lesquels vous êtes responsable des configurations de sécurité des composants Kubernetes.
- Fournit un moyen standard approuvé par la communauté de vous assurer que vous avez configuré votre cluster et vos nœuds Kubernetes en toute sécurité lors de l'utilisation d'Amazon EKS.
- Se compose de quatre sections : configuration de journalisation du plan de contrôle, configurations de sécurité des nœuds, politiques et services managés.
- Prend en charge toutes les versions de Kubernetes actuellement disponibles dans Amazon EKS et peut être exécuté à l'aide de kube-bench, un outil open source standard pour vérifier la configuration à l'aide de l'évaluation CIS sur les clusters Kubernetes.

Pour en savoir plus, consultez Présentation de la norme CIS Amazon EKS.

Versions de la plateforme Amazon EKS

Les versions de la plateforme Amazon EKS représentent les fonctionnalités du plan de contrôle du cluster, notamment les indicateurs de serveur d'KubernetesAPI activés et la version actuelle du Kubernetes correctif. Les nouveaux clusters sont déployés avec la dernière version de la plateforme. Pour plus de détails, consultez Versions de la plateforme Amazon EKS.

Vous pouvez mettre à jour un cluster Amazon EKS vers de nouvelles versions de Kubernetes. Lorsque de nouvelles versions Kubernetes sont mises à disposition dans Amazon EKS, nous vous recommandons de mettre à jour, de manière proactive, vos clusters et d'utiliser la dernière version disponible. Pour plus d'informations sur les versions de Kubernetes dans EKS, consultez Versions Kubernetes Amazon EKS.

Liste des vulnérabilités du système d'exploitation

Liste des vulnérabilités AL2023

Suivez les événements liés à la sécurité ou à la confidentialité d'Amazon Linux 2023 dans le centre de Linux sécurité Amazon ou abonnez-vous au flux RSS associé. Les événements de sécurité et de confidentialité incluent une présentation du problème, des packages et des instructions relatives à la mise à jour de vos instances pour résoudre le problème.

Liste des vulnérabilités d'Amazon Linux 2

Suivez les événements liés à la sécurité ou à la confidentialité d'Amazon Linux 2 dans le centre de Linux sécurité Amazon ou abonnez-vous au flux RSS associé. Les événements de sécurité et de confidentialité incluent une présentation du problème, des packages et des instructions relatives à la mise à jour de vos instances pour résoudre le problème.

Détection de nœuds avec Amazon Inspector

Vous pouvez utiliser Amazon Inspector pour vérifier l'accessibilité réseau indésirable de vos nœuds et les vulnérabilités de ces instances Amazon EC2.

Détection de clusters et de nœuds avec Amazon GuardDuty

Service de détection des GuardDuty menaces Amazon qui aide à protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. Parmi les autres fonctionnalités, GuardDuty propose les deux fonctionnalités suivantes qui détectent les menaces potentielles pour vos clusters EKS : la protection EKS et la surveillance du temps d'exécution.

Pour plus d'informations, voir Détectez les menaces avec Amazon GuardDuty.

Bonnes pratiques de sécurité pour Amazon EKS

Les bonnes pratiques de sécurité pour Amazon EKS sont maintenues sur Github : https://aws.github.io/aws-eks-best-practices/security/docs/

Politique de sécurité de pod

Le contrôleur d'admission de la stratégie de sécurité de Pod Kubernetes valide les demandes de création et de mise à jour de Pod en fonction d'un ensemble de règles. Par défaut, les clusters Amazon EKS sont livrés avec une politique de sécurité entièrement permissive sans aucune

restriction. Pour plus d'informations, consultez Stratégies de sécurité de Pod dans la documentation Kubernetes.

> **Note**
>
> `PodSecurityPolicy` (PSP) est devenue obsolète dans Kubernetes `1.21` et a été supprimée dans Kubernetes `1.25`. Les PSPs sont remplacés par Pod Security Admission (PSA), un contrôleur d'admission intégré qui met en œuvre les contrôles de sécurité décrits dans les normes de sécurité des pods (PSS). PSA et PSS ont tous deux atteint des états de fonctionnalités bêta et sont activés par défaut dans Amazon EKS. Pour remédier à la suppression de PSP dans la version `1.25`, nous vous recommandons d'implémenter PSS dans Amazon EKS. Pour plus d'informations, consultez la section Implémentation des normes de sécurité des pods dans Amazon EKS sur le blog AWS.

Stratégie de sécurité de Pod Amazon EKS par défaut

Les clusters Amazon EKS avec Kubernetes version `1.13` ou ultérieure ont une stratégie de sécurité de Pod par défaut nommée `eks.privileged`. Cette stratégie n'a aucune limite en ce qui concerne le type de Pod pouvant être accepté dans le système, ce qui équivaut à exécuter Kubernetes avec le contrôleur `PodSecurityPolicy` désactivé.

> **Note**
>
> Cette politique a été créée pour maintenir la rétrocompatibilité avec les clusters pour lesquels le contrôleur `PodSecurityPolicy` n'était pas activé. Vous pouvez créer des politiques plus restrictives pour votre cluster et pour les espaces de noms et comptes de service individuels, puis supprimer la politique par défaut pour activer les politiques plus restrictives.

Vous pouvez afficher la politique par défaut à l'aide de la commande suivante.

```
kubectl get psp eks.privileged
```

L'exemple qui suit illustre un résultat.

```
NAME              PRIV    CAPS    SELINUX    RUNASUSER    FSGROUP    SUPGROUP
 READONLYROOTFS   VOLUMES
```

```
eks.privileged   true   *      RunAsAny   RunAsAny    RunAsAny   RunAsAny   false
      *
```

Pour plus de détails, vous pouvez décrire la politique avec la commande suivante.

```
kubectl describe psp eks.privileged
```

L'exemple qui suit illustre un résultat.

```
Name:  eks.privileged

Settings:
  Allow Privileged:                       true
  Allow Privilege Escalation:             0xc0004ce5f8
  Default Add Capabilities:               <none>
  Required Drop Capabilities:             <none>
  Allowed Capabilities:                   *
  Allowed Volume Types:                   *
  Allow Host Network:                     true
  Allow Host Ports:                       0-65535
  Allow Host PID:                         true
  Allow Host IPC:                         true
  Read Only Root Filesystem:              false
  SELinux Context Strategy: RunAsAny
    User:                                 <none>
    Role:                                 <none>
    Type:                                 <none>
    Level:                                <none>
  Run As User Strategy: RunAsAny
    Ranges:                               <none>
  FSGroup Strategy: RunAsAny
    Ranges:                               <none>
  Supplemental Groups Strategy: RunAsAny
    Ranges:                               <none>
```

Vous pouvez consulter le fichier YAML complet pour connaître la stratégie de sécurité du Pod `eks.privileged`, son rôle de cluster et la liaison du rôle de cluster dans Installer ou restaurer la stratégie de sécurité de Pod par défaut.

Supprimer la stratégie de sécurité de Pod Amazon EKS par défaut

Si vous créez des stratégies plus restrictives pour vos Pods, vous pouvez, après l'avoir fait, supprimer la stratégie de sécurité par défaut du Pod `eks.privileged` d'Amazon EKS pour activer vos stratégies personnalisées.

> **Important**
>
> Si vous utilisez la version `1.7.0` ou une version ultérieure du plug-in CNI et que vous attribuez une politique de sécurité de Pod personnalisée au compte de service `aws-node` Kubernetes utilisé pour les Pods `aws-node` déployés par le Daemonset, la politique doit comporter `NET_ADMIN` dans sa section `allowedCapabilities`, avec `hostNetwork: true` et `privileged: true` dans les `spec` de la politique.

Pour supprimer la stratégie de sécurité de Pod par défaut

1. Créez un fichier nommé *`privileged-podsecuritypolicy.yaml`* avec le contenu du fichier d'exemple dans Installer ou restaurer la stratégie de sécurité de Pod par défaut.
2. Supprimez le fichier YAML avec la commande suivante. Ceci supprime la stratégie de sécurité de Pod par défaut, le `ClusterRole` et le `ClusterRoleBinding` qui lui est associé.

```
kubectl delete -f privileged-podsecuritypolicy.yaml
```

Installer ou restaurer la stratégie de sécurité de Pod par défaut

Si vous effectuez une mise à niveau à partir d'une version antérieure de Kubernetes ou si vous avez modifié ou supprimé la stratégie de sécurité de Pod par défaut `eks.privileged` Amazon EKS, vous pouvez la restaurer en procédant comme suit.

Pour installer ou restaurer la stratégie de sécurité de Pod par défaut

1. Créez un fichier nommé *`privileged-podsecuritypolicy.yaml`* avec le contenu suivant.

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: eks.privileged
```

```
  annotations:
    kubernetes.io/description: 'privileged allows full unrestricted access to
      Pod features, as if the PodSecurityPolicy controller was not enabled.'
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  volumes:
  - '*'
  hostNetwork: true
  hostPorts:
  - min: 0
    max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
  readOnlyRootFilesystem: false

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:podsecuritypolicy:privileged
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
rules:
- apiGroups:
  - policy
  resourceNames:
  - eks.privileged
  resources:
```

```
  - podsecuritypolicies
  verbs:
  - use

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:podsecuritypolicy:authenticated
  annotations:
    kubernetes.io/description: 'Allow all authenticated users to create privileged
 Pods.'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:podsecuritypolicy:privileged
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
```

2. Appliquez le fichier YAML avec la commande suivante.

```
kubectl apply -f privileged-podsecuritypolicy.yaml
```

FAQ sur la suppression de la politique de sécurité des pods (PSP)

`PodSecurityPolicy` est devenue obsolète dans Kubernetes `1.21` et a été supprimée dans Kubernetes `1.25`. Si vous l'utilisez PodSecurityPolicy dans votre cluster, vous devez migrer vers les normes de sécurité Kubernetes Pod intégrées (PSS) ou vers une policy-as-code solution avant de passer à la version de votre cluster **`1.25`** afin d'éviter toute interruption de vos charges de travail. Sélectionnez une question fréquemment posée pour en savoir plus.

Qu'est-ce qu'un PSP ?

PodSecurityPolicyest un contrôleur d'admission intégré qui permet à un administrateur de cluster de contrôler les aspects des spécifications sensibles à la sécurité. Pod Si un Pod répond aux exigences

de la PSP, ce Pod est admis dans le cluster comme d'habitude. Si un Pod ne répond pas aux exigences de la PSP, ce Pod est rejeté et ne peut pas s'exécuter.

La suppression de PSP est-elle spécifique à Amazon EKS ou est-elle effectuée en amont dans Kubernetes ?

Il s'agit d'une modification en amont du projet Kubernetes, et non d'une modification apportée dans Amazon EKS. La PSP est devenue obsolète dans Kubernetes `1.21` et a été supprimée dans Kubernetes `1.25`. La communauté Kubernetes a identifié de graves problèmes d'utilisation avec la PSP. Ces problèmes concernaient notamment l'octroi accidentel d'autorisations plus étendues que prévu et la difficulté d'inspecter celles que la PSPs applique dans une situation donnée. La résolution de ces problèmes nécessitait des changements radicaux. C'est la principale raison pour laquelle la communauté Kubernetes a décidé de supprimer la PSP.

Comment m'assurer que j'utilise la PSPs dans mes clusters Amazon EKS ?

Pour vous assurer que vous utilisez la PSPs dans votre cluster, vous pouvez exécuter la commande suivante :

```
kubectl get psp
```

Pour afficher les Pods concernés par la PSPs dans votre cluster, exécutez la commande suivante. Cette commande renvoie le nom du Pod, l'espace de noms et les PSPs :

```
kubectl get pod -A -o jsonpath='{range.items[?(@.metadata.annotations.kubernetes
\.io/psp)]}{.metadata.name}{"\t"}{.metadata.namespace}{"\t"}
{.metadata.annotations.kubernetes\.io/psp}{"\n"}'
```

Que faire si j'utilise la PSPs dans mon cluster Amazon EKS ?

Avant de procéder à la mise à niveau de votre cluster vers la version `1.25`, vous devez migrer votre PSPs vers l'une des alternatives suivantes :

- Kubernetes PSS.

- olicy-as-code Solutions P issues de l'Kubernetesenvironnement.

En réponse à l'abandon de la PSP et à la nécessité permanente de contrôler la sécurité du Pod dès le départ, la communauté Kubernetes a créé une solution intégrée avec (PSS) et Pod Security Admission (PSA). Le webhook PSA implémente les contrôles définis dans PSS.

Vous pouvez consulter les bonnes pratiques pour la migration de la PSPs vers les PSS intégrées dans le Guide des bonnes pratiques EKS. Nous vous recommandons également de consulter notre article sur l'implémentation des normes de sécurité des pods dans Amazon EKS. Les références supplémentaires incluent Migrate from PodSecurityPolicy to the built-in PodSecurity Admission Controller et Mapping PodSecurityPolicies to Pod Security Standards.

olicy-as-code Les solutions P fournissent des garde-fous pour guider les utilisateurs du cluster et prévenir les comportements indésirables grâce à des contrôles automatisés prescrits. Les olicy-as-code solutions P utilisent généralement les contrôleurs d'admission dynamiques Kubernetes pour intercepter le flux de demandes du serveur d'KubernetesAPI à l'aide d'un appel webhook. olicy-as-code Les solutions P modifient et valident les charges utiles des demandes en fonction de politiques écrites et stockées sous forme de code.

Plusieurs policy-as-code solutions open source sont disponibles pourKubernetes. Pour consulter les meilleures pratiques en matière de migration PSPs vers une policy-as-code solution, consultez la olicy-as-code section P de la page Pod Security sur GitHub.

Je vois qu'une PSP a appelé **`eks.privileged`** dans mon cluster. De quoi s'agit-il et que puis-je faire ?

Les clusters Amazon EKS exécutant Kubernetes `1.13` ou une version ultérieure sont dotés d'une PSP par défaut nommée `eks.privileged`. Cette politique est créée dans les clusters `1.24` et les clusters de version antérieure. Elle n'est pas utilisée dans les clusters `1.25` et les clusters de version ultérieure. Amazon EKS migre automatiquement cette PSP vers une application basée sur PSS. Aucune action de votre part n'est nécessaire.

Amazon EKS apportera-t-il des modifications aux PSPs déjà présentes dans mon cluster lorsque je mettrai à jour mon cluster vers la version **`1.25`** ?

Non. Outre `eks.privileged`, qui est une PSP créée par Amazon EKS, aucune modification n'est apportée aux autresPSPs de votre cluster lors de la mise à niveau vers la version `1.25`.

Amazon EKS empêchera-t-il une mise à jour du cluster vers la version `1.25` si je n'ai pas encore effectué de migration depuis ma PSP ?

Non, Amazon EKS n'empêchera pas la mise à jour du cluster vers la version `1.25` si vous n'avez pas encore effectué de migration depuis votre PSP.

Que se passe-t-il si j'oublie de migrer PSPs vers PSS/PSA ou vers une policy-as-code solution avant de mettre à jour la version de mon cluster `1.25` ? Puis-je migrer après avoir mis à jour mon cluster ?

Lorsqu'un cluster contenant une PSP est mis à niveau vers Kubernetes `1.25`, le serveur d'API ne reconnaît pas la ressource PSP dans la version `1.25`. Des étendues de sécurité incorrectes peuvent alors être attribuées aux Pods. Pour une liste exhaustive des implications, voir PodSecurityPolicy Migrer depuis le contrôleur PodSecurity d'admission intégré.

Quel est l'impact de ce changement sur la sécurité des pods pour les charges de travail Windows ?

Nous ne prévoyons aucun impact spécifique sur les charges de travail Windows. PodSecurityContext possède un champ appelé `windowsOptions` dans l'`PodSpec v1`API pour WindowsPods. Cela utilise PSS dans Kubernetes `1.25`. Pour plus d'informations, et pour découvrir les meilleures pratiques concernant l'application de la PSS pour des charges de travail Windows, consultez le Guide des bonnes pratiques EKS et la documentation Kubernetes.

Utilisation des secrets AWS Secrets Manager avec Kubernetes

Pour afficher les secrets de Secrets Manager et les paramètres de Parameter Store en tant que fichiers montés dans des Pods Amazon EKS, vous pouvez utiliser ASCP (AWS Secrets and Configuration Provider) pour le pilote CSI Kubernetes Secrets Store.

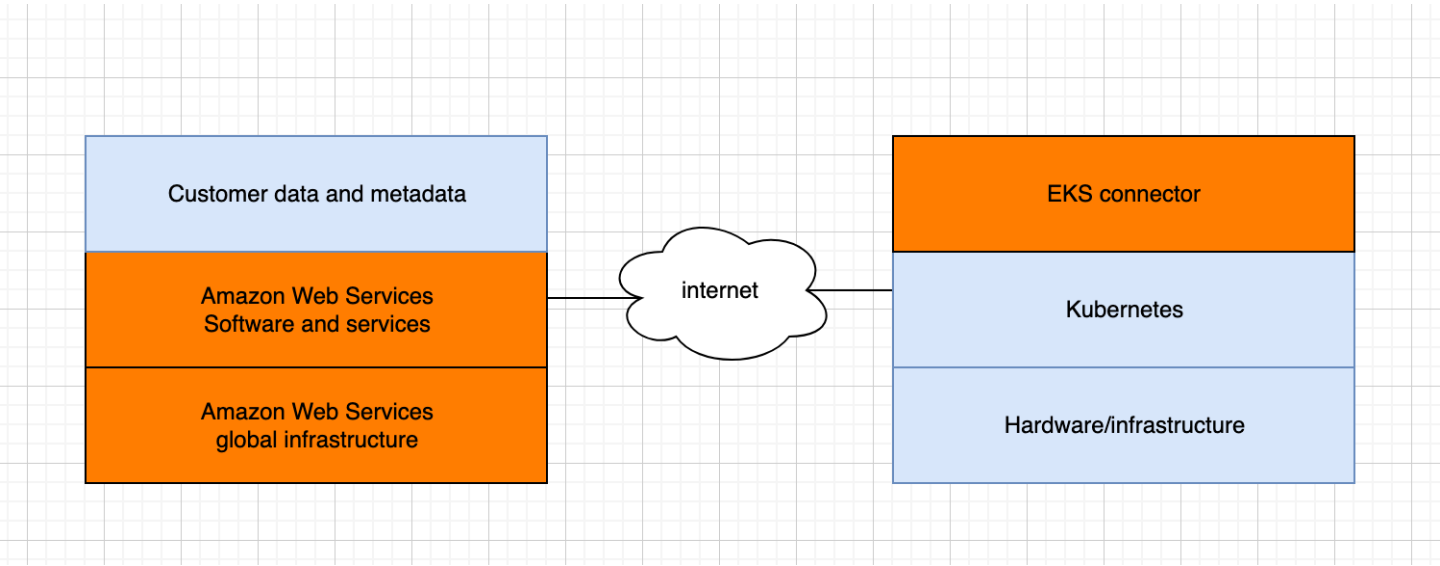
Avec l'ASCP, vous pouvez stocker et gérer vos secrets dans Secrets Manager, puis les récupérer via vos applications exécutées sur Amazon EKS. Vous pouvez utiliser des rôles et des politiques IAM pour limiter l'accès à vos secrets à des Pods Kubernetes spécifiques dans un cluster. L'ASCP récupère l'identité du Pod et l'échange contre un rôle IAM. ASCP assume le rôle IAM du Pod, puis il peut récupérer les secrets de Secrets Manager autorisés pour ce rôle.

Si vous utilisez la rotation automatique de Secrets Manager pour vos secrets, vous pouvez également utiliser la fonction de réconciliation de rotation du pilote CSI de Secrets Store pour vous assurer que vous récupérez le dernier secret de Secrets Manager.

Pour plus d'informations, consultez Utilisation des secrets de Secrets Manager dans Amazon EKS dans le Guide de l'utilisateur AWS Secrets Manager.

Considérations relatives à Amazon EKS Connector

Amazon EKS Connector est un composant open source qui s'exécute sur votre cluster Kubernetes. Ce cluster peut être situé en dehors de l'environnement AWS. Cela crée des considérations supplémentaires pour les responsabilités en matière de sécurité. Cette configuration peut être illustrée par le schéma suivant. L'orange représente les responsabilités AWS, et le bleu représente les responsabilités des clients :



Cette rubrique décrit les différences entre le modèle de responsabilité si le cluster connecté est externe à AWS.

Responsabilités AWS

- Maintenir, créer et fournir Amazon EKS Connector, qui est un composant open source qui s'exécute sur le cluster Kubernetes d'un client et communique avec AWS.
- Maintenir la sécurité des communications de la couche de transport et de la couche d'application entre le cluster Kubernetes connecté et les services AWS.

Responsabilités client

- Sécurité spécifique au cluster Kubernetes, notamment dans les lignes suivantes :
  - Les secrets Kubernetes doivent être correctement chiffrés et protégés.

  - Verrouillez l'accès à l'espace de noms `eks-connector`.
- Configurer des autorisations de contrôle d'accès basé sur les rôles (RBAC) pour gérer l'accès des principaux IAM depuis AWS. Pour obtenir des instructions, consultez Octroi d'un accès à un principal IAM pour afficher les ressources Kubernetes sur un cluster.
- Installation et mise à niveau d'Amazon EKS Connector.
- Maintenir le matériel, les logiciels et l'infrastructure prenant en charge le cluster Kubernetes connecté.
- Sécuriser leurs comptes AWS (par exemple, en utilisant des informations d'identification d'utilisateur root sécurisées).

Afficher les ressources Kubernetes

Vous pouvez afficher les ressources Kubernetes déployées sur votre cluster avec la AWS Management Console. Vous ne pouvez pas afficher Kubernetes les ressources avec le AWS CLI ou `eksctl`. Pour afficher les ressources Kubernetes à l'aide d'un outil de ligne de commande, utilisez `kubectl`.

Prérequis

Pour afficher l'onglet Ressources et la section Nœuds de l'onglet Compute du AWS Management Console, le principal IAM que vous utilisez doit disposer d'un IAM et Kubernetes d'autorisations spécifiques. Pour plus d'informations, consultez Autorisations nécessaires.

Pour consulter les Kubernetes ressources à l'aide du AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Dans la liste Clusters, sélectionnez le cluster qui contient les ressources Kubernetes que vous souhaitez afficher.
3. Sélectionnez l'onglet Resources (Ressources).
4. Sélectionnez un groupe Resource type (Type de ressource) pour lequel vous souhaitez afficher les ressources, comme Workloads (Charges de travail). Vous voyez une liste des types de ressources dans ce groupe.
5. Sélectionnez un type de ressource, tel que Deployments (Déploiements), dans le groupe Workloads (Charges de travail). Vous voyez une description du type de ressource, un lien vers la documentation Kubernetes pour plus d'informations sur le type de ressource et une liste des ressources de ce type qui sont déployées sur votre cluster. Si la liste est vide, aucune ressource de ce type n'est déployée sur votre cluster.
6. Sélectionnez une ressource pour afficher plus d'informations à son sujet. Essayez les exemples suivants :

    - Sélectionnez le groupe Workloads (Charges de travail), sélectionnez le type de ressource Deployments (Déploiements), puis sélectionnez la ressource coredns. Lorsque vous sélectionnez une ressource, vous êtes dans Structured view (Vue structurée), par défaut. Pour certains types de ressources, vous voyez une section Pod dans Structured view (Vue structurée). Cette section répertorie les Pods gérés par la charge de travail. Vous pouvez

sélectionner n'importe quel Pod répertorié pour afficher les informations sur le Pod. Tous les types de ressources n'affichent pas d'informations dans Structured View (Vue structurée). Si vous sélectionnez Vue brute dans le coin supérieur droit de la page de la ressource, vous voyez la réponse JSON complète de l'API Kubernetes pour la ressource.

- Sélectionnez le groupe Cluster, puis sélectionnez le type de ressource Nodes (Nœuds). Vous voyez une liste de tous les nœuds de votre cluster. Les nœuds peuvent être n'importe quel type de nœud Amazon EKS. Il s'agit de la même liste que celle que vous voyez dans la section Nodes (Nœuds) lorsque vous sélectionnez l'onglet Compute (Calcul) pour votre cluster. Sélectionnez une ressource de nœud dans la liste. Dans Structured view (Vue structurée), vous voyez également une section Pod. Cette section vous montre tous les Pods en cours d'exécution sur le nœud.

Autorisations nécessaires

Pour afficher l'onglet Ressources et la section Nœuds de l'onglet Compute du AWS Management Console, le principal IAM que vous utilisez doit disposer d'un IAM et Kubernetes d'autorisations minimales spécifiques. Procédez comme suit pour accorder les autorisations requises à vos principaux IAM.

1. Assurez-vous que les autorisations `eks:AccessKubernetesApi`, ainsi que les autres autorisations IAM nécessaires pour afficher les ressources Kubernetes, sont attribuées au principal IAM que vous utilisez. Pour plus d'informations sur la modification des autorisations d'un principal IAM, consultez la rubrique Contrôle de l'accès des principaux du Guide de l'utilisateur IAM. Pour plus d'informations sur la modification des autorisations pour un rôle, consultez Modification d'une politique d'autorisations de rôle (console) dans le guide de l'utilisateur IAM.

   L'exemple de politique suivant inclut les autorisations nécessaires pour qu'un principal puisse afficher les ressources Kubernetes de tous les clusters de votre compte. Remplacez *111122223333* par votre ID de compte AWS .

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "eks:ListFargateProfiles",
                   "eks:DescribeNodegroup",
   ```

```
                "eks:ListNodegroups",
                "eks:ListUpdates",
                "eks:AccessKubernetesApi",
                "eks:ListAddons",
                "eks:DescribeCluster",
                "eks:DescribeAddonVersions",
                "eks:ListClusters",
                "eks:ListIdentityProviderConfigs",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:GetParameter",
            "Resource": "arn:aws:ssm:*:111122223333:parameter/*"
        }
    ]
}
```

Pour afficher les nœuds des clusters connectés, le rôle IAM du connecteur Amazon EKS doit être en mesure d'emprunter l'identité du principal dans le cluster. Cela permet au Amazon EKS Connector de mapper le principal à un utilisateur Kubernetes.

2. Créez une `rolebinding` ou une `clusterrolebinding` Kubernetes qui est liée à un `role` ou un `clusterrole` Kubernetesou qui dispose des autorisations nécessaires pour afficher les ressources Kubernetes. Pour en savoir plus sur les rôles et les liaisons de rôles Kubernetes, consultez Utilisation de l'autorisation RBAC dans la documentation Kubernetes. Vous pouvez appliquer l'un des manifestes suivants à votre cluster qui crée un `role` et une `rolebinding` ou un `clusterrole` et une `clusterrolebinding` avec les autorisations Kubernetes nécessaires :

   Afficher les ressources Kubernetes dans tous les espaces de noms

   Le nom du groupe dans le fichier est `eks-console-dashboard-full-access-group`. Appliquez le manifeste à votre cluster à l'aide de la commande suivante :

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
   ```

Afficher les ressources Kubernetes dans un espace de noms spécifique

L'espace de noms de ce fichier est `default`. Le nom du groupe dans le fichier est `eks-console-dashboard-restricted-access-group`. Appliquez le manifeste à votre cluster à l'aide de la commande suivante :

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

Si vous devez modifier le nom du groupe Kubernetes, l'espace de noms, les autorisations ou toute autre configuration dans le fichier, téléchargez le fichier et modifiez-le avant de l'appliquer à votre cluster :

1. Téléchargez le fichier avec l'une des commandes suivantes :

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

2. Modifiez le fichier si nécessaire.
3. Appliquez le manifeste à votre cluster avec l'une des commandes suivantes :

```
kubectl apply -f eks-console-full-access.yaml
```

```
kubectl apply -f eks-console-restricted-access.yaml
```

3. Mappez le principal IAM à l'utilisateur ou au groupe Kubernetes dans la `ConfigMap` `aws-auth`. Vous pouvez utiliser un outil tel que `eksctl` pour mettre à jour le `ConfigMap` ou vous pouvez le mettre à jour manuellement en le modifiant.

> **Important**
>
> Nous vous recommandons d'utiliser `eksctl`, ou un autre outil, pour modifier le `ConfigMap`. Pour plus d'informations sur les autres outils que vous pouvez utiliser, consultez Utiliser des outils pour apporter des modifications au `aws-auth` `ConfigMap`

dans les guides des bonnes pratiques Amazon EKS. Un `aws-auth` `ConfigMap` mis en forme incorrectement peut entraîner la perte de l'accès à votre cluster.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

1. Affichez les mappages actuels dans le `ConfigMap`. Remplacez *`my-cluster`* par le nom de votre cluster. *`region-code`*Remplacez-le par Région AWS celui dans lequel se trouve votre cluster.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

L'exemple qui suit illustre un résultat.

```
ARN
                  USERNAME                                GROUPS
            ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. Ajoutez un mappage pour un rôle. Cet exemple suppose que vous avez attaché les autorisations IAM lors de la première étape à un rôle nommé *`my-console-viewer-role`*. Remplacez *`111122223333`* par votre ID de compte.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:role/my-console-viewer-role \
    --group eks-console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> L'ARN de rôle ne peut pas inclure de chemin d'accès tel que `role/my-team/developers/my-role`. Le format de l'ARN doit être `arn:aws:iam::111122223333:role/my-role`. Dans cet exemple, `my-team/developers/` doit être supprimé.

L'exemple qui suit illustre un résultat.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-console-viewer-role" to auth ConfigMap
```

3. Ajoutez un mappage pour un utilisateur. Les bonnes pratiques IAM recommandent d'accorder des autorisations à des rôles plutôt qu'à des utilisateurs. Cet exemple suppose que vous avez attaché les autorisations IAM lors de la première étape à un utilisateur nommé *my-user*. Remplacez *111122223333* par votre ID de compte.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user \
    --group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

L'exemple qui suit illustre un résultat.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-user" to auth ConfigMap
```

4. Affichez de nouveau les mappages dans le `ConfigMap`.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

L'exemple qui suit illustre un résultat.

```
ARN
                    USERNAME                                      GROUPS
                      ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/my-console-viewer-role
                                                                  eks-console-
dashboard-full-access-group
arn:aws:iam::111122223333:user/my-user
                                                                  eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

Pour plus d'informations sur l'ajout d'utilisateurs ou de rôles IAM à la `ConfigMap` `aws-auth`, consultez la rubrique Ajout de principaux IAM à votre cluster Amazon EKS.

1. Ouvrez le `aws-auth` `ConfigMap` pour le modifier.

```
kubectl edit -n kube-system configmap/aws-auth
```

2. Ajoutez les mappages à `aws-auth` `ConfigMap`, mais ne remplacez aucun des mappages existants. L'exemple suivant ajoute des mappages entre les principaux IAM avec les autorisations ajoutées lors de la première étape et les groupes Kubernetes créés à l'étape précédente :

   - Le rôle *my-console-viewer-role* et le `eks-console-dashboard-full-access-group`.
   - L'utilisateur *my-user* et le `eks-console-dashboard-restricted-access-group`.

   Ces exemples supposent que vous avez attaché les autorisations IAM au cours de la première étape à un rôle nommé *my-console-viewer-role* et à un utilisateur nommé *my-user*. Remplacez *111122223333* par votre identifiant de AWS compte.

```
apiVersion: v1
data:
mapRoles: |
  - groups:
    - eks-console-dashboard-full-access-group
```

```
    rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
    username: my-console-viewer-role
mapUsers: |
  - groups:
    - eks-console-dashboard-restricted-access-group
    userarn: arn:aws:iam::111122223333:user/my-user
    username: my-user
```

> **Important**
>
> L'ARN de rôle ne peut pas inclure de chemin d'accès tel que `role/my-team/developers/my-console-viewer-role`. Le format de l'ARN doit être `arn:aws:iam::111122223333:role/my-console-viewer-role`. Dans cet exemple, `my-team/developers/` doit être supprimé.

3. Enregistrez le fichier et quittez votre éditeur de texte.

Observabilité dans Amazon EKS

Vous pouvez observer vos données dans Amazon EKS à l'aide de nombreux outils de surveillance ou de journalisation disponibles. Les données de vos journaux Amazon EKS peuvent être diffusées vers Services AWS ou vers des outils partenaires à des fins d'analyse des données. De nombreux services sont disponibles dans le document AWS Management Console qui fournissent des données pour résoudre vos problèmes avec Amazon EKS. Vous pouvez également utiliser une solution open source AWS prise en charge pour surveiller l'infrastructure Amazon EKS.

Vous pouvez afficher l'état et les détails du cluster en sélectionnant le nom de votre cluster après avoir sélectionné Clusters dans le volet de navigation de gauche de la console Amazon EKS. Pour afficher des détails sur les ressources Kubernetes existantes déployées sur votre cluster, consultez Afficher les ressources Kubernetes.

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon EKS et de vos AWS solutions. Nous vous recommandons de collecter des données de surveillance à partir de toutes les parties de votre AWS solution. De cette façon, vous pouvez déboguer plus facilement une éventuelle défaillance à plusieurs points. Avant de commencer à surveiller Amazon EKS, assurez-vous que votre plan de surveillance réponde aux questions suivantes.

- Quels sont vos objectifs ? Avez-vous besoin de notifications en temps réel si vos clusters évoluent considérablement ?
- Quelles ressources faut-il observer ?
- À quelle fréquence devez-vous observer ces ressources ? Votre entreprise veut-elle réagir rapidement aux risques ?
- Quels outils avez-vous l'intention d'utiliser ? Si vous l'exécutez déjà dans AWS Fargate le cadre de votre lancement, vous pouvez utiliser le routeur de journalisation intégré.
- À qui souhaitez-vous confier les tâches de surveillance ?
- À qui voulez-vous que les notifications soient envoyées en cas de problème ?

Journalisation et surveillance dans Amazon EKS

Amazon EKS fournit des outils intégrés pour la journalisation et la surveillance. La journalisation du plan de contrôle enregistre tous les appels d'API vers vos clusters, les informations d'audit capturant

quels utilisateurs ont effectué quelles actions sur vos clusters, et les informations basées sur les rôles. Pour plus d'informations, consultez la section Journalisation et surveillance sur Amazon EKS dans les directives prescriptives AWS .

La journalisation du plan de contrôle Amazon EKS fournit des journaux d'audit et de diagnostic directement depuis le plan de contrôle Amazon EKS vers CloudWatch les journaux de votre compte. Ces journaux vous permettent de sécuriser et d'exécuter facilement vos clusters. Vous pouvez sélectionner les types de journaux exacts dont vous avez besoin, et les journaux sont envoyés sous forme de flux de journaux à un groupe pour chaque cluster Amazon EKS inclus CloudWatch. Pour plus d'informations, consultez Journalisation de plan de contrôle d'Amazon EKS.

> **Note**
>
> Lorsque vous consultez les journaux de l'authentificateur Amazon EKS sur Amazon CloudWatch, les entrées affichées contiennent du texte similaire à l'exemple de texte suivant.
>
> ```
> level=info msg="mapping IAM role" groups="[]"
>  role="arn:aws:iam::111122223333:role/XXXXXXXXXXXXXXXXXX-
> NodeManagerRole-XXXXXXXX" username="eks:node-manager"
> ```
>
> Des entrées contenant ce texte sont attendues. `username` est un rôle de service interne Amazon EKS qui effectue des opérations spécifiques pour les groupes de nœuds gérés et Fargate.
> Pour une journalisation personnalisable de bas niveau, la journalisation Kubernetes est disponible.

Amazon EKS est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amazon EKS. CloudTrail capture tous les appels d'API pour Amazon EKS sous forme d'événements. Ces captures incluent les appels de la console Amazon EKS et les appels de code vers les opérations d'API Amazon EKS. Pour plus d'informations, consultez Journalisation des appels d'API Amazon EKS avec AWS CloudTrail.

Le serveur d'API Kubernetes expose un certain nombre de métriques qui sont utiles pour la surveillance et l'analyse. Pour plus d'informations, consultez Prometheus métriques .

Fluent BitPour configurer des Amazon CloudWatch journaux personnalisés, consultez la section Configuration Fluent Bit dans le guide de CloudWatch l'utilisateur Amazon.

Outils de journalisation et de surveillance d'Amazon EKS

Amazon Web Services fournit différents outils que vous pouvez utiliser pour surveiller Amazon EKS. Vous pouvez configurer certains outils pour configurer la surveillance automatique, mais certains nécessitent des appels manuels. Nous vous recommandons d'automatiser les tâches de surveillance autant que votre environnement et le jeu d'outils existant le permettent.

Outils de journalisation

Zones	Outil	Description	Configuration
Applications	Informations sur les CloudWatch conteneurs Amazon	Il collecte, regroupe et récapitule les métriques et les journaux de vos applications et microservices conteneurisés.	Procédures de configuration
Plan de contrôle	AWS CloudTrail	Il consigne les appels d'API par un utilisateur, un rôle ou un service.	Procédures de configuration
Plusieurs zones pour les AWS Fargate instances	AWS Fargate routeur à journaux	Par AWS Fargate exemple, il diffuse les journaux vers AWS des services ou des outils partenaires. Utilisations d'AWS pour Fluent Bit. Les journaux peuvent	Procédures de configuration

Zones	Outil	Description	Configuration
		être diffusés vers d'autres outils Services AWS ou vers des outils partenaires.	

Outils de supervision

Zones	Outil	Description	Configuration
Applications	CloudWatch Container Insights	CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices.	Procédures de configuration
Applications	AWS Distro pour OpenTelemetry (ADOT)	Il collecte et envoie des métriques corrélées, des données de suivi et des métadonnées aux services de AWS surveillance ou aux partenaires. Il peut être	Procédures de configuration

Zones	Outil	Description	Configuration
		configuré via CloudWatch Container Insights.	
Applications	Amazon DevOps Guru	Il détecte les performances opérationnelles et la disponibilité au niveau des nœuds.	Procédures de configuration
Applications	AWS X-Ray	Il reçoit des données de suivi concernant votre application. Ces données de suivi incluent les demandes entrantes et sortantes ainsi que les métadonnées relatives aux demandes. Pour Amazon EKS, l'implémentation nécessite le module complémentaire OpenTelemetry.	Procédures de configuration

Zones	Outil	Description	Configuration
Applications	Opérateur Amazon CloudWatch Observability	L'opérateur Amazon CloudWatch Observability collecte des métriques, des journaux et des données de suivi. Il les envoie à Amazon CloudWatch et AWS X-Ray.	Procédures de configuration
Plan de contrôle	Prometheus	CloudWatch Les taux d'ingestion des journaux, de stockage des archives et d'analyse des données s'appliquent aux journaux du plan de contrôle activés.	Procédures de configuration

Prometheus métriques

Prometheus est une base de données de surveillance et de séries temporelles qui récupère ces données à partir des points de terminaison. Elle permet d'interroger, d'agréger et de stocker les données collectées. Vous pouvez également l'utiliser pour les alertes et l'agrégation des alertes. Cette rubrique explique comment configurer Prometheus en tant qu'option gérée ou open source. La surveillance des métriques du plan de contrôle d'Amazon EKS est un cas d'utilisation courant.

Amazon Managed Service for Prometheus est un service de surveillance et d'alerte compatible avec Prometheus qui facilite la surveillance des applications et infrastructures conteneurisées à grande échelle. Il s'agit d'un service entièrement géré qui met automatiquement à l'échelle l'ingestion, le stockage, l'interrogation et l'alerte de vos métriques. Il s'intègre également aux services AWS de sécurité pour permettre un accès rapide et sécurisé à vos données. Vous pouvez utiliser le langage de requête open source PromQL pour interroger vos métriques et émettre des alertes à leur sujet.

Pour plus d'informations sur l'utilisation des métriques Prometheus après les avoir activées, consultez le Guide de l'utilisateur Amazon Managed Service for Prometheus.

Activation des métriques Prometheus lors de la création d'un cluster

> **Important**
>
> Les ressources Amazon Managed Service for Prometheus ne font pas partie du cycle de vie du cluster et doivent être gérées indépendamment du cluster. Lorsque vous supprimez votre cluster, assurez-vous de supprimer également tous les scrapers applicables afin de mettre fin aux coûts applicables. Pour plus d'informations, consultez la section Rechercher et supprimer des scrapers dans le guide de l'utilisateur d'Amazon Managed Service for Prometheus.

Lorsque vous créez un nouveau cluster, vous pouvez activer l'option d'envoi des métriques à Prometheus. Dans le AWS Management Console, cette option se trouve dans l'étape Configurer l'observabilité qui consiste à créer un nouveau cluster. Pour plus d'informations, consultez Création d'un cluster Amazon EKS.

Prometheus découvre et collecte les métriques de votre cluster par le biais d'un modèle basé sur le pull appelé scraping. Les scrapers sont configurés pour collecter des données à partir de l'infrastructure de votre cluster et des applications conteneurisées.

Lorsque vous activez l'option d'envoi des métriques Prometheus, Amazon Managed Service for Prometheus fournit un scraper sans agent entièrement géré. Utilisez les options de Configuration avancée suivantes pour personnaliser le scraper par défaut selon vos besoins.

Alias du scraper

(Facultatif) Saisissez un alias unique pour le scraper.

Destination

Choisissez un espace de travail Amazon Managed Service for Prometheus. Un espace de travail est un espace logique dédié au stockage et à l'interrogation des métriques Prometheus. Grâce à cet espace de travail, vous pourrez consulter les métriques Prometheus sur l'ensemble des comptes qui y ont accès. L'option Créer un nouvel espace de travail indique à Amazon EKS de créer un espace de travail en votre nom à l'aide de l'Alias d'espace de travail que vous fournissez. L'option Sélectionner un espace de travail existant vous permet de sélectionner un espace de travail existant dans une liste déroulante. Pour plus d'informations sur les espaces de travail, consultez Gestion des espaces de travail dans le Guide de l'utilisateur Amazon Managed Service for Prometheus.

Accès à un service

Cette section résume les autorisations que vous accordez lors de l'envoi de métriques Prometheus :

- Autoriser Amazon Managed Service for Prometheus à décrire le cluster Amazon EKS récupéré
- Autoriser l'écriture à distance dans l'espace de travail Amazon Managed Prometheus

Si `AmazonManagedScraperRole` existe déjà, le scraper l'utilise. Choisissez le lien `AmazonManagedScraperRole` pour voir les Détails de l'autorisation. Si `AmazonManagedScraperRole` n'existe pas encore, choisissez le lien Afficher les détails des autorisations pour voir les autorisations spécifiques que vous accordez en envoyant des métriques Prometheus.

Sous-réseaux

Affichez les sous-réseaux dont le scraper héritera. Si vous devez les modifier, revenez à l'étape de création du cluster Spécifier la mise en réseau.

Groupes de sécurité

Affichez les groupes de sécurité dont le scraper héritera. Si vous devez les modifier, revenez à l'étape de création du cluster Spécifier la mise en réseau.

Configuration du scraper

Modifiez la configuration du scraper au format YAML si nécessaire. Pour ce faire, utilisez le formulaire ou chargez un fichier YAML de remplacement. Pour plus d'informations, consultez Configuration du scraper dans le Guide de l'utilisateur Amazon Managed Service for Prometheus.

Amazon Managed Service for Prometheus fait référence au scraper sans agent créé aux côtés du cluster en tant que collecteur géré AWS . Pour plus d'informations sur les collecteurs AWS gérés, consultez la section Collecteurs AWS gérés dans le guide de l'utilisateur d'Amazon Managed Service for Prometheus.

> **Important**
>
> Vous devez configurer votre `aws-auth ConfigMap` pour donner au scraper des autorisations dans le cluster. Pour plus d'informations, consultez Configuration de votre cluster Amazon EKS dans le Guide de l'utilisateur Amazon Managed Service for Prometheus.

Affichage des détails du scraper Prometheus

Après avoir créé un cluster avec l'option de métriques Prometheus activée, vous pouvez afficher les détails de votre scraper Prometheus. Lorsque vous visualisez votre cluster dans le AWS Management Console, choisissez l'onglet Observabilité. Un tableau affiche la liste des scrappers du cluster, avec des informations telles que l'ID de scraper, l'alias, le statut et la date de création.

Pour obtenir plus de détails sur le scraper, cliquez sur un lien d'ID de scraper. Par exemple, vous pouvez afficher la configuration du scraper, l'Amazon Resource Name (ARN), l'URL d'écriture à distance et les informations de mise en réseau. Vous pouvez utiliser l'ID de scraper comme entrée dans Amazon Managed Service pour les opérations API Prometheus comme `DescribeScraper` et `DeleteScraper`. Vous pouvez également utiliser l'API pour créer d'autres scrapeurs.

Pour plus d'informations sur l'utilisation de l'API Prometheus, consultez la Référence de l'API Amazon Managed Service for Prometheus.

Déploiement de Prometheus à l'aide de Helm

Vous pouvez également déployer Prometheus dans votre cluster avec Helm V3. Si vous avez déjà installé Helm, vous pouvez vérifier votre version avec la commande `helm version`. Helm est un gestionnaire de packages pour les clusters Kubernetes. Pour de plus amples informations sur Helm et son installation, veuillez consulter Utilisation de Helm avec Amazon EKS.

Après avoir configuré Helm pour votre cluster Amazon EKS, vous pouvez l'utiliser pour déployer Prometheus avec les étapes suivantes.

Pour déployer Prometheus à l'aide de Helm

1. Créez un espace de noms Prometheus.

```
kubectl create namespace prometheus
```

2. Ajoutez le référentiel de graphiques `prometheus-community`.

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
```

3. Déployez Prometheus.

```
helm upgrade -i prometheus prometheus-community/prometheus \
    --namespace prometheus \
    --set alertmanager.persistence.storageClass="gp2" \
    --set server.persistentVolume.storageClass="gp2"
```

> **Note**
>
> Si vous obtenez l'erreur `Error: failed to download "stable/prometheus" (hint: running `helm repo update` may help)` lors de l'exécution de cette commande, exécutez `helm repo update prometheus-community`, puis réessayez d'exécuter la commande Étape 2.
> Si vous obtenez l'erreur `Error: rendered manifests contain a resource that already exists`, exécutez `helm uninstall` *`your-release-name`* `-n` *`namespace`*, puis réessayez d'exécuter la commande Étape 3.

4. Vérifiez que tous les Pods dans l'espace de noms `prometheus` sont à l'état `READY`.

```
kubectl get pods -n prometheus
```

L'exemple qui suit illustre un résultat.

```
NAME                                             READY   STATUS    RESTARTS   AGE
prometheus-alertmanager-59b4c8c744-r7bgp         1/2     Running   0          48s
prometheus-kube-state-metrics-7cfd87cf99-jkz2f   1/1     Running   0          48s
prometheus-node-exporter-jcjqz                   1/1     Running   0          48s
prometheus-node-exporter-jxv2h                   1/1     Running   0          48s
prometheus-node-exporter-vbdks                   1/1     Running   0          48s
```

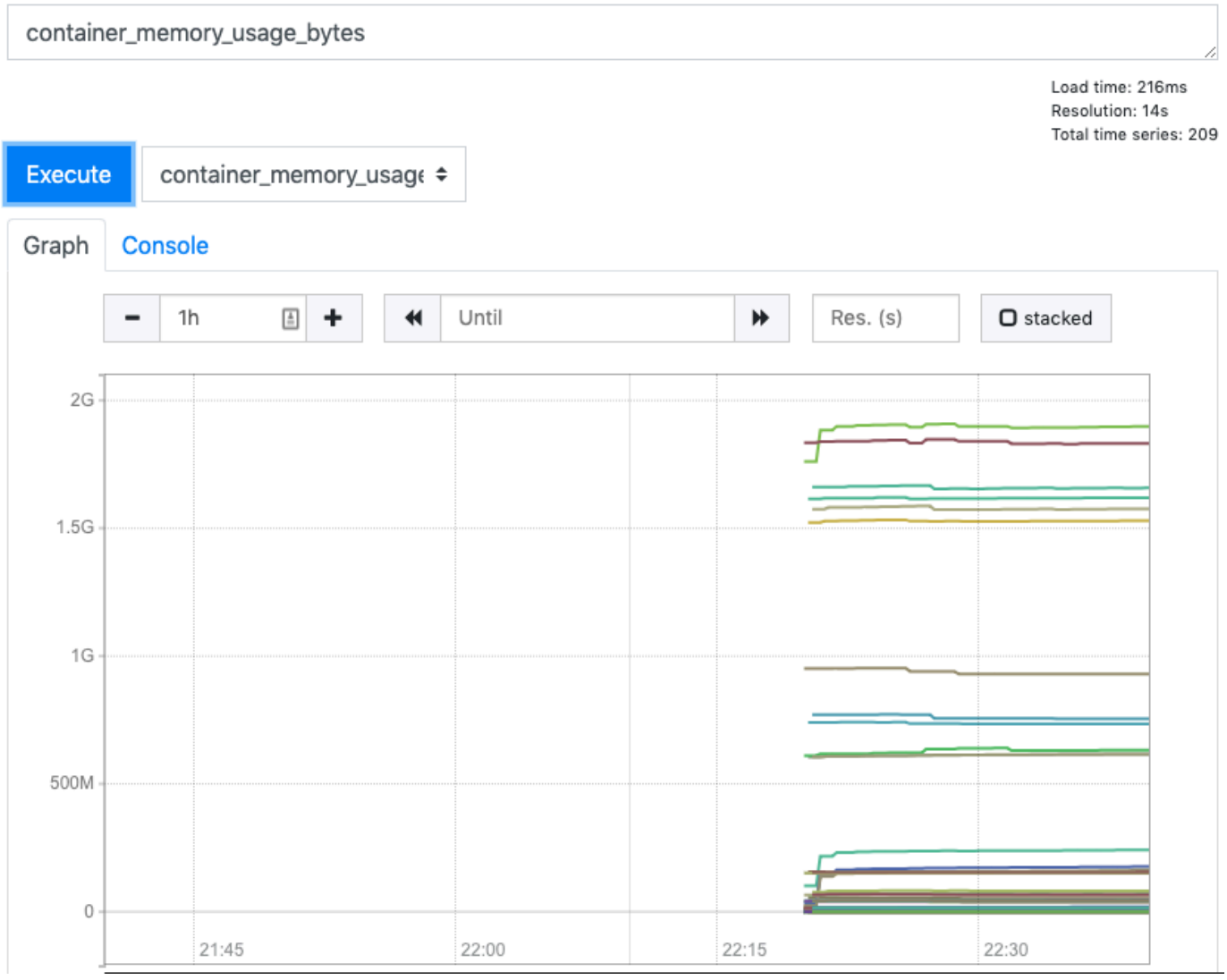
```
prometheus-pushgateway-76c444b68c-82tnw     1/1     Running   0          48s
prometheus-server-775957f748-mmht9          1/2     Running   0          48s
```

5. Utilisez `kubectl` pour transférer la console Prometheus vers votre ordinateur local.

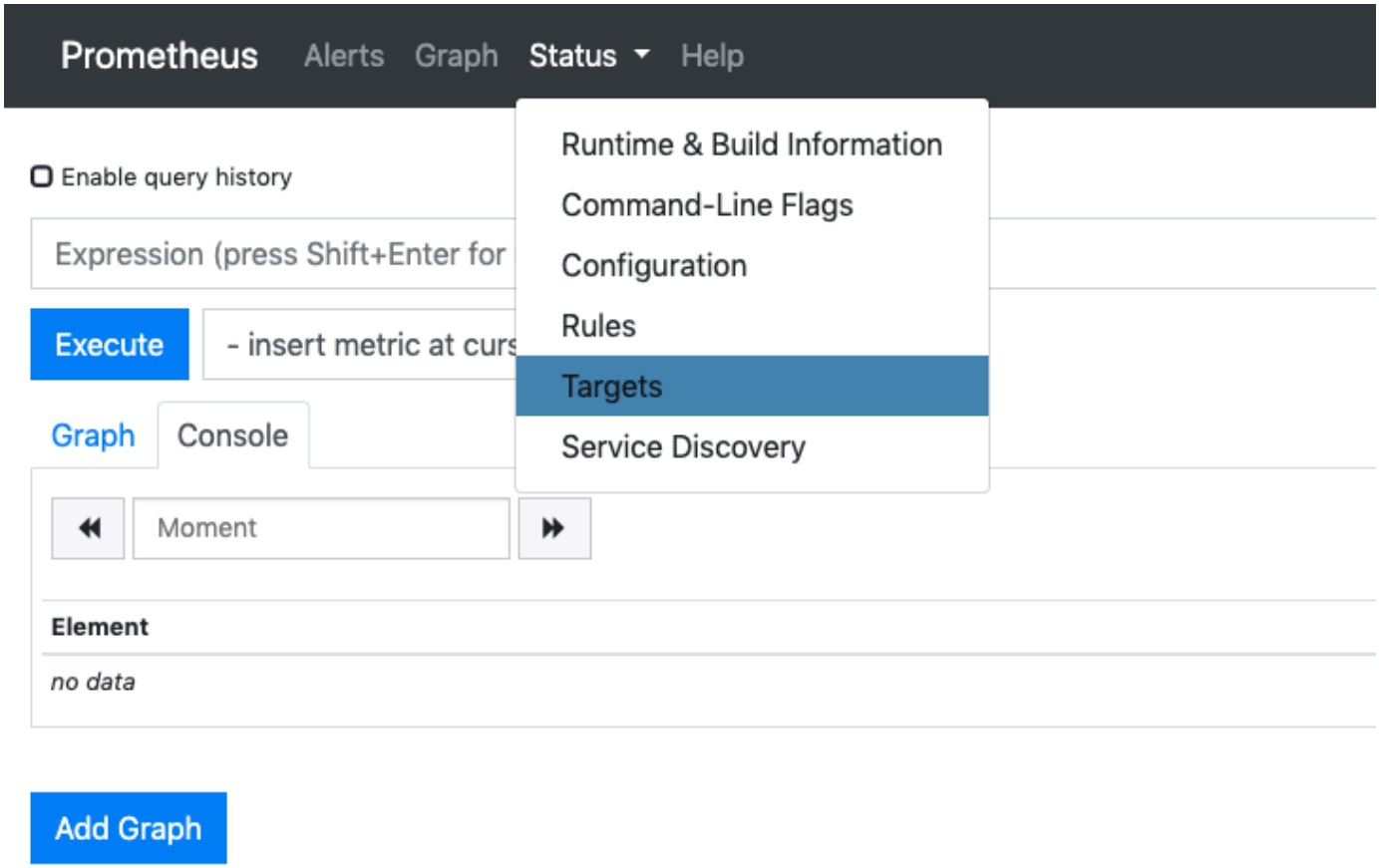
```
kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
```

6. Faites pointer un navigateur web sur `http://localhost:9090` pour afficher la console Prometheus.

7. Choisissez une métrique dans le menu - insert metric at cursor (insérer une métrique sur le curseur), puis choisissez Execute (Exécuter). Choisissez l'onglet Graph (Graphique) pour afficher la métrique au fil du temps. L'image suivante montre `container_memory_usage_bytes` au fil du temps.



8. Dans la barre de navigation supérieure, choisissez Status (Statut), puis Targets (Cibles).



Tous les points de terminaison Kubernetes connectés à Prometheus à l'aide de la découverte de service sont affichés.

Affichage des métriques brutes du plan de contrôle

Comme alternative au déploiement de Prometheus, le serveur d'API Kubernetes expose un certain nombre de métriques qui sont représentées dans un format Prometheus. Ces métriques sont utiles pour la surveillance et l'analyse. Elles sont exposées en interne via un point de terminaison des métriques qui renvoie à l'API HTTP `/metrics`. Comme pour les autres points de terminaison, ce point de terminaison est exposé sur le plan de contrôle Amazon EKS. Ce point de terminaison est principalement utile pour examiner une métrique spécifique. Pour analyser les métriques dans le temps, nous vous recommandons de déployer Prometheus.

Pour afficher la sortie des métriques brutes, utilisez `kubectl` avec l'indicateur `--raw`. Cette commande vous permet de transmettre n'importe quel chemin HTTP et renvoie la réponse brute.

```
kubectl get --raw /metrics
```

L'exemple qui suit illustre un résultat.

```
[...]
# HELP rest_client_requests_total Number of HTTP requests, partitioned by status code,
 method, and host.
# TYPE rest_client_requests_total counter
rest_client_requests_total{code="200",host="127.0.0.1:21362",method="POST"} 4994
rest_client_requests_total{code="200",host="127.0.0.1:443",method="DELETE"} 1
rest_client_requests_total{code="200",host="127.0.0.1:443",method="GET"} 1.326086e+06
rest_client_requests_total{code="200",host="127.0.0.1:443",method="PUT"} 862173
rest_client_requests_total{code="404",host="127.0.0.1:443",method="GET"} 2
rest_client_requests_total{code="409",host="127.0.0.1:443",method="POST"} 3
rest_client_requests_total{code="409",host="127.0.0.1:443",method="PUT"} 8
# HELP ssh_tunnel_open_count Counter of ssh tunnel total open attempts
# TYPE ssh_tunnel_open_count counter
ssh_tunnel_open_count 0
# HELP ssh_tunnel_open_fail_count Counter of ssh tunnel failed open attempts
# TYPE ssh_tunnel_open_fail_count counter
ssh_tunnel_open_fail_count 0
```

Cette sortie brute renvoie tel quel ce que le serveur d'API expose. Les différentes métriques sont répertoriées par ligne, chaque ligne comprenant un nom de métrique, des balises et une valeur.

```
metric_name{"tag"="value"[,...]}
            value
```

Support des modules complémentaires Amazon EKS pour Amazon CloudWatch

Amazon CloudWatch Observability collecte des journaux, des métriques et des données de traçage en temps réel. Il les envoie à Amazon CloudWatch et AWS X-Ray. Vous pouvez installer ce module complémentaire pour activer à la fois les signaux CloudWatch d'application et CloudWatch Container Insights améliorer l'observabilité d'Amazon EKS. Cela vous aide à surveiller l'état et les performances de votre infrastructure et de vos applications conteneurisées. Amazon CloudWatch Observability Operator est conçu pour installer et configurer les composants nécessaires.

Amazon EKS prend en charge Amazon CloudWatch Observability Operator en tant que module complémentaire Amazon EKS. Le module complémentaire autorise à Container Insights la fois les nœuds Linux Windows et les nœuds de travail du cluster. Pour être Container Insights

activéeWindows, la version du module complémentaire Amazon EKS doit être `1.5.0` ou supérieure. Actuellement, CloudWatch Application Signals n'est pas pris en charge sur Amazon EKSWindows.

Les rubriques ci-dessous décrivent comment commencer à utiliser Amazon CloudWatch Observability Operator pour votre cluster Amazon EKS.

- Pour obtenir des instructions sur l'installation de ce module complémentaire, consultez Installer l' CloudWatch agent à l'aide des modules complémentaires Amazon EKS d' CloudWatch Observability dans le guide de l' CloudWatch utilisateur Amazon.
- Pour plus d'informations sur les signaux CloudWatch d'application, consultez la section Signaux d'application.
- Pour plus d'informationsContainer Insights, consultez la section Utilisation Container Insights dans le guide de CloudWatch l'utilisateur Amazon.

Journalisation de plan de contrôle d'Amazon EKS

La journalisation du plan de contrôle Amazon EKS fournit des journaux d'audit et de diagnostic directement depuis le plan de contrôle Amazon EKS vers CloudWatch les journaux de votre compte. Ces journaux vous permettent de sécuriser et d'exécuter facilement vos clusters. Vous pouvez sélectionner les types de journaux exacts dont vous avez besoin, et les journaux sont envoyés sous forme de flux de journaux à un groupe pour chaque cluster Amazon EKS inclus CloudWatch. Pour plus d'informations, consultez Amazon CloudWatch logging.

Vous pouvez commencer à utiliser la journalisation de plan de contrôle d'Amazon EKS en choisissant les types de journal que vous voulez activer pour chaque cluster Amazon EKS nouveau ou existant. Vous pouvez activer ou désactiver chaque type de journal cluster par cluster à l'aide de la AWS Management Console, de la AWS CLI (version `1.16.139` ou supérieure) ou de l'API Amazon EKS. Lorsque cette option est activée, les journaux sont automatiquement envoyés depuis le cluster Amazon EKS aux CloudWatch journaux du même compte.

Lorsque vous utilisez la journalisation de plan de contrôle d'Amazon EKS, des frais standards Amazon EKS vous sont facturés pour chaque cluster que vous exécutez. Les frais standard d'ingestion et de stockage des données CloudWatch Logs vous sont facturés pour tous les journaux envoyés à CloudWatch Logs depuis vos clusters. Vous sont facturées également les ressources AWS, comme les instances Amazon EC2 ou les volumes Amazon EBS, que vous allouez dans le cadre de votre cluster.

Les types de journal de plan de contrôle de cluster suivants sont disponibles. Chaque type de journal correspond à un composant du plan de contrôle Kubernetes. Pour en savoir plus sur ces composants, consultez Composants Kubernetes dans la documentation Kubernetes.

Serveur d'API (**`api`**)

Le serveur d'API de votre cluster est le composant de plan de contrôle qui expose l'API Kubernetes. Si vous activez les journaux du serveur d'API au moment du lancement du cluster, ou peu après, les journaux incluent les indicateurs qui ont été utilisés pour démarrer le serveur d'API. Pour plus d'informations, consultez `kube-apiserver` et la stratégie d'audit dans la documentation Kubernetes.

Audit (**`audit`**)

Les journaux d'audit Kubernetes fournissent un enregistrement des différents utilisateurs, administrateurs ou composants système qui affectent votre cluster. Pour plus d'informations, consultez la section Audit dans la documentation Kubernetes.

Authentificateur (**`authenticator`**)

Les journaux d'authentification sont propres à Amazon EKS. Ces journaux représentent le composant de plan de contrôle utilisé par Amazon EKS pour l'authentification du contrôle d'accès basé sur les rôles (RBAC) de Kubernetes à l'aide es informations d'identification IAM. Pour plus d'informations, consultez Gestion du cluster.

Gestionnaire de contrôleur (**`controllerManager`**)

Le gestionnaire de contrôleur gère les boucles de contrôle de base qui sont fournies avec Kubernetes. Pour plus d'informations, consultez la section kube-controller-managerdans la documentation Kubernetes.

Planificateur (**`scheduler`**)

Le composant planificateur gère quand et où les Pods s'exécutent dans votre cluster. Pour plus d'informations, consultez kube-scheduler dans la documentation Kubernetes.

Activation et désactivation des journaux de plan de contrôle

Par défaut, les journaux du plan de contrôle du cluster ne sont pas envoyés à CloudWatch Logs. Vous devez activer chaque type de journal individuellement pour envoyer des journaux pour votre cluster. CloudWatch Les taux d'ingestion des journaux, de stockage des archives et d'analyse des

données s'appliquent aux journaux du plan de contrôle activés. Pour plus d'informations, consultez CloudWatch les tarifs.

Pour mettre à jour la configuration de la journalisation du plan de contrôle, Amazon EKS a besoin d'un maximum de cinq adresses IP disponibles dans chaque sous-réseau. Lorsque vous activez un type de journal, les journaux sont envoyés avec un niveau de détail de journal 2.

AWS Management Console

Pour activer ou désactiver des journaux de plan de contrôle à l'aide de l'AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez le nom du cluster pour afficher les informations le concernant.
3. Sélectionnez l'onglet Observabilité.
4. Dans la section Journalisation du plan de contrôle, sélectionnez Gérer la journalisation.
5. Pour chaque type de journal, choisissez si celui-ci doit être activé ou désactivé. Par défaut, chaque type de journal est désactivé.
6. Choisissez Enregistrer les modifications pour terminer.

AWS CLI

Pour activer ou désactiver des journaux de plan de contrôle à l'aide de l'AWS CLI

1. Vous pouvez vérifier la version de votre AWS CLI à l'aide de la commande suivante.

```
aws --version
```

   Si la version de votre AWS CLI est inférieure à `1.16.139`, vous devez d'abord la mettre à jour vers la version la plus récente. Pour installer ou mettre à niveau l'interface AWS CLI, voir Installation de AWS Command Line Interface dans le Guide de l'utilisateur AWS Command Line Interface.
2. Mettez à jour la configuration d'exportation de journal de plan de contrôle de votre cluster à l'aide de la commande AWS CLI suivante. Remplacez *`my-cluster`* par le nom de votre cluster et spécifiez les valeurs d'accès au point de terminaison souhaitées.

> **Note**
>
> La commande suivante envoie tous les types de journaux disponibles à CloudWatch Logs.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

L'exemple qui suit illustre un résultat.

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "InProgress",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

3. Surveillez l'état de la mise à jour de votre configuration de journal avec la commande suivante, en indiquant le nom de votre cluster et l'ID de mise à jour qui ont été renvoyés par la commande précédente. Votre mise à jour est terminée lorsqu'elle affiche l'état `Successful`.

```
aws eks describe-update \
    --region region-code\
    --name my-cluster \
```

```
    --update-id 883405c8-65c6-4758-8cee-2a7c1340a6d9
```

L'exemple qui suit illustre un résultat.

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "Successful",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

Affichage des journaux de plan de contrôle de cluster

Une fois que vous avez activé l'un des types de journaux du plan de contrôle pour votre cluster Amazon EKS, vous pouvez les consulter sur la CloudWatch console.

Pour en savoir plus sur l'affichage, l'analyse et la gestion des connexions CloudWatch, consultez le guide de l'utilisateur Amazon CloudWatch Logs.

Pour consulter les journaux du plan de contrôle de votre cluster sur la CloudWatch console

1. Ouvrez la CloudWatch console. Ce lien ouvre la console et affiche vos groupes de journaux disponibles actuels et les filtre avec le préfixe `/aws/eks`.
2. Choisissez le cluster pour lequel vous souhaitez afficher les journaux. Le format du nom de groupe de journaux est `/aws/eks/my-cluster/cluster`.
3. Choisissez le flux de journal à afficher. La liste suivante décrit le format de nom de flux du journal pour chaque type de journal.

> **Note**
>
> À mesure que les données de flux de journal augmentent, les noms de flux de journal font l'objet d'une rotation. Lorsqu'il existe plusieurs flux de journaux pour un type de journal particulier, vous pouvez afficher le dernier flux en recherchant son nom avec l'heure du dernier événement (Last event time).

- Journaux de composant de serveur d'API Kubernetes (**api**) : `kube-apiserver-1234567890abcdef01234567890abcde`
- Audit (**audit**) : `kube-apiserver-audit-1234567890abcdef01234567890abcde`
- Authentificateur (**authenticator**) : `authenticator-1234567890abcdef01234567890abcde`
- Gestionnaire de contrôleur (**controllerManager**) : `kube-controller-manager-1234567890abcdef01234567890abcde`
- Planificateur (**scheduler**) : `kube-scheduler-1234567890abcdef01234567890abcde`

4. Parcourez les événements du flux de journaux.

   Par exemple, vous devez voir les indicateurs de serveur d'API initiaux correspondant au cluster lorsque vous consultez le haut de `kube-apiserver-1234567890abcdef01234567890abcde`.

   > **Note**
   >
   > Si vous ne voyez pas les journaux du serveur d'API au début du flux de journaux, il est probable que le fichier journal du serveur d'API a été l'objet d'une rotation sur le serveur avant d'activer la journalisation du serveur API sur le serveur. Les fichiers journaux qui font l'objet d'une rotation avant que la journalisation du serveur d'API ne soit activée ne peuvent pas être exportés vers CloudWatch.
   > Toutefois, vous pouvez créer un nouveau cluster avec la même version de Kubernetes et activer la journalisation du serveur API lorsque vous créez le cluster. Les clusters avec la même version de plate-forme ont les mêmes indicateurs activés, donc vos indicateurs doivent correspondre aux indicateurs du nouveau cluster. Lorsque vous avez fini de visualiser les drapeaux du nouveau cluster dans CloudWatch, vous pouvez supprimer le nouveau cluster.

Journalisation des appels d'API Amazon EKS avec AWS CloudTrail

Amazon EKS est intégré à AWS CloudTrail. CloudTrail est un service qui fournit un enregistrement des actions effectuées par un utilisateur, un rôle ou un service AWS dans Amazon EKS. CloudTrail capture tous les appels d'API pour Amazon EKS en tant qu'événements. Cela inclut les appels de la console Amazon EKS et les appels de code aux opérations de l'API Amazon EKS.

Si vous créez un journal d'activité, vous pouvez activer la livraison continue d'événements CloudTrail à un compartiment Amazon S3, y compris des événements pour Amazon S3. Cela inclut les événements pour Amazon EKS. Si vous ne configurez pas de journal d'activité, vous pouvez toujours afficher les événements les plus récents dans la console CloudTrail dans Event history (Historique des événements). Grâce aux informations collectées par CloudTrail, vous pouvez déterminer plusieurs détails sur une demande. Par exemple, vous pouvez déterminer quand la demande a été envoyée à Amazon EKS, l'adresse IP à partir de laquelle la demande a été effectuée et qui a effectué la demande.

Pour en savoir plus sur CloudTrail, consultez le AWS CloudTrailGuide de l'utilisateur .

Rubriques

- Informations relatives à Amazon EKS dans CloudTrail
- Présentation des entrées des fichiers journaux Amazon EKS
- Activer la collecte des métriques de groupe Auto Scaling

Informations relatives à Amazon EKS dans CloudTrail

Lorsque vous créez votre compte AWS, CloudTrail est également activé dans votre compte AWS. Lorsqu'une activité se produit dans Amazon EKS, cette activité est enregistrée dans un événement CloudTrail avec d'autres événements de service AWS dans Event history (Historique des événements). Vous pouvez afficher, rechercher et télécharger les événements récents dans votre compte AWS. Pour plus d'informations, consultez Affichage des événements avec l'historique des événements CloudTrail.

Pour enregistrer en continu des événements de votre compte AWS, y compris les événements pour Amazon EKS, créez un journal d'activité. Un journal d'activité permet à CloudTrail de distribuer les fichiers journaux vers Simple Storage Service (Amazon S3) bucket. Par défaut, lorsque vous créez un journal d'activité dans la console, il s'applique à toutes les régions Régions AWS. Le journal de suivi consigne les événements de toutes les Régions AWS dans la partition AWS et transfère les fichiers

journaux dans le compartiment Amazon S3 de votre choix. De plus, vous pouvez configurer d'autres services AWS pour analyser plus en profondeur les données d'événement collectées dans les journaux CloudTrail et agir sur celles-ci. Pour plus d'informations, veuillez consulter les ressources suivantes.

- Présentation de la création d'un journal d'activité
- Intégrations et services pris en charge par CloudTrail
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers journaux CloudTrail de plusieurs régions et Réception de fichiers journaux CloudTrail de plusieurs comptes

Toutes les actions Amazon EKS sont consignées par CloudTrail et documentées dans la référence d'API Amazon EKS. Par exemple, les appels aux sections `CreateCluster`, `ListClusters` et `DeleteCluster` génèrent des entrées dans les fichiers journaux CloudTrail.

Chaque événement ou entrée de journal contient des détails sur le type d'identité IAM ayant effectué la demande, ainsi que les informations d'identification qui ont été utilisées. Si des informations d'identification temporaires ont été utilisées, l'entrée montre comment ces informations d'identification ont été obtenues.

Pour plus d'informations, consultez l'élément userIdentity CloudTrail.

Présentation des entrées des fichiers journaux Amazon EKS

Un journal d'activité est une configuration qui permet d'envoyer des événements sous forme de fichiers journaux à un compartiment Simple Storage Service (Amazon S3) que vous spécifiez. Les fichiers journaux CloudTrail peuvent contenir une ou plusieurs entrées. Un événement représente une demande émise par une source et comprend des informations sur l'action demandée. Cela comprend des informations comme la date et l'heure de l'action et les paramètres de demande qui ont été utilisés. Les fichiers journaux CloudTrail ne constituent pas une série ordonnée retraçant les appels d'API publiques. Ils ne suivent aucun ordre précis.

L'exemple suivant montre une entrée de journal CloudTrail qui illustre l'action `CreateCluster`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
```

```
    "arn": "arn:aws:iam::111122223333:user/username",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "username"
  },
  "eventTime": "2018-05-28T19:16:43Z",
  "eventSource": "eks.amazonaws.com",
  "eventName": "CreateCluster",
  "awsRegion": "region-code",
  "sourceIPAddress": "205.251.233.178",
  "userAgent": "PostmanRuntime/6.4.0",
  "requestParameters": {
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-a670c2df",
        "subnet-4f8c5004"
      ]
    },
    "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
    "clusterName": "test"
  },
  "responseElements": {
    "cluster": {
      "clusterName": "test",
      "status": "CREATING",
      "createdAt": 1527535003.208,
      "certificateAuthority": {},
      "arn": "arn:aws:eks:region-code:111122223333:cluster/test",
      "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
      "version": "1.10",
      "resourcesVpcConfig": {
        "securityGroupIds": [],
        "vpcId": "vpc-21277358",
        "subnetIds": [
          "subnet-a670c2df",
          "subnet-4f8c5004"
        ]
      }
    }
  },
  "requestID": "a7a0735d-62ab-11e8-9f79-81ce5b2b7d37",
  "eventID": "eab22523-174a-499c-9dd6-91e7be3ff8e3",
```

```
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Entrées de journal des rôles liés à un service Amazon EKS

Les rôles liés à un service Amazon EKS effectuent des appels d'API vers les ressources AWS. Des entrées de journal CloudTrail avec `username: AWSServiceRoleForAmazonEKS` et `username: AWSServiceRoleForAmazonEKSNodegroup` apparaissent pour des appels effectués par les rôles liés à un service Amazon EKS. Pour plus d'informations sur Amazon EKS et les rôles liés aux services, consultez Utilisation des rôles liés à un service pour Amazon EKS.

L'exemple suivant présente une entrée de journal CloudTrail qui illustre une action `DeleteInstanceProfile` effectuée par le rôle lié à un service `AWSServiceRoleForAmazonEKSNodegroup`, notée dans le `sessionContext`.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3WHGPEZ7SJ2CW55C5:EKS",
        "arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForAmazonEKSNodegroup/EKS",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA3WHGPEZ7SJ2CW55C5",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/eks-
nodegroup.amazonaws.com/AWSServiceRoleForAmazonEKSNodegroup",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAmazonEKSNodegroup"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-02-26T00:56:33Z"
            }
        },
        "invokedBy": "eks-nodegroup.amazonaws.com"
```

```
    },
    "eventTime": "2020-02-26T00:56:34Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "DeleteInstanceProfile",
    "awsRegion": "region-code",
    "sourceIPAddress": "eks-nodegroup.amazonaws.com",
    "userAgent": "eks-nodegroup.amazonaws.com",
    "requestParameters": {
        "instanceProfileName": "eks-11111111-2222-3333-4444-abcdef123456"
    },
    "responseElements": null,
    "requestID": "11111111-2222-3333-4444-abcdef123456",
    "eventID": "11111111-2222-3333-4444-abcdef123456",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

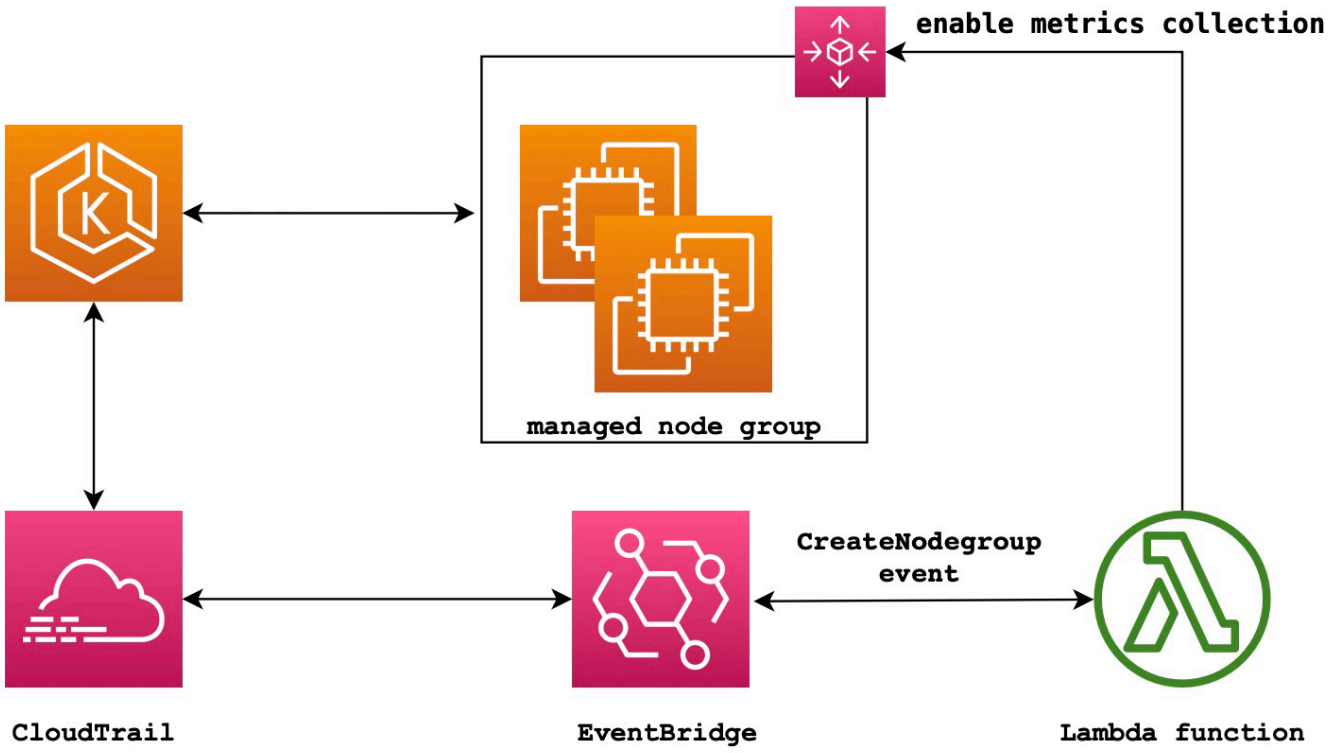
Activer la collecte des métriques de groupe Auto Scaling

Cette rubrique explique comment activer la collecte des métriques de groupe Auto Scaling à l'aide de AWS Lambda et AWS CloudTrail. Amazon EKS n'active pas automatiquement la collecte des métriques de groupe Auto Scaling créées pour les nœuds gérés.

Vous pouvez utiliser les métriques de groupe Auto Scaling pour suivre les modifications apportées à un groupe Auto Scaling et pour définir des alarmes sur les valeurs de seuil. Les métriques de groupe Auto Scaling sont disponibles dans la console Auto Scaling ou sur la CloudWatch console Amazon. Une fois activé, le groupe Auto Scaling envoie des échantillons de données à Amazon CloudWatch toutes les minutes. L'activation de ces métriques est gratuite.

En activant la collecte des métriques de groupe Auto Scaling, vous pourrez surveiller le dimensionnement des groupes de nœuds gérés. Les métriques de groupe Auto Scaling indiquent les tailles minimale, maximale et souhaitée d'un groupe Auto Scaling. Vous pouvez créer une alarme si le nombre de nœuds d'un groupe de nœuds est inférieur à la taille minimale, ce qui indique que le groupe est défectueux. Le suivi de la taille des groupes de nœuds permet également d'ajuster le nombre maximal afin que votre plan de données ne se retrouve pas à court de capacité.

Lorsque vous créez un groupe de nœuds gérés, AWS CloudTrail envoie un `CreateNodegroup` événement à Amazon EventBridge. En créant une EventBridge règle Amazon correspondant à l'`CreateNodegroup`événement, vous déclenchez une fonction Lambda pour permettre la collecte de métriques de groupe pour le groupe Auto Scaling associé au groupe de nœuds gérés.



Pour activer la collecte des métriques de groupe Auto Scaling

1. Créez un rôle IAM pour Lambda.

```
LAMBDA_ROLE=$(aws iam create-role \
  --role-name lambda-asg-enable-metrics \
  --assume-role-policy-document '{"Version": "2012-10-17","Statement":
 [{ "Effect": "Allow", "Principal": {"Service": "lambda.amazonaws.com"}, "Action":
 "sts:AssumeRole"}]}' \
  --output text \
  --query 'Role.Arn')
echo $LAMBDA_ROLE
```

2. Créez une stratégie permettant de décrire les groupes de nœuds Amazon EKS et d'activer la collecte des métriques de groupe Auto Scaling.

```
cat > /tmp/lambda-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
```

```
      {
        "Effect": "Allow",
        "Action": [
            "eks:DescribeNodegroup",
            "autoscaling:EnableMetricsCollection"
            ],
        "Resource": [
            "*"
        ]
      }
    ]
}
EOF
LAMBDA_POLICY_ARN=$(aws iam create-policy \
  --policy-name lambda-asg-enable-metrics-policy \
  --policy-document file:///tmp/lambda-policy.json \
  --output text \
  --query 'Policy.Arn')
echo $LAMBDA_POLICY_ARN
```

3. Attachez la stratégie au rôle IAM pour Lambda.

```
aws iam attach-role-policy \
  --policy-arn $LAMBDA_POLICY_ARN \
  --role-name lambda-asg-enable-metrics
```

4. Ajoutez la politique `AWSLambdaBasicExecutionRole` gérée, qui dispose des autorisations dont la fonction a besoin pour écrire des journaux dans CloudWatch Logs.

```
aws iam attach-role-policy \
  --role-name lambda-asg-enable-metrics \
  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
```

5. Créez le code Lambda.

```
cat > /tmp/lambda-handler.py <<EOF
import json
import boto3
import time
import logging

eks = boto3.client('eks')
autoscaling = boto3.client('autoscaling')
```

```
logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    ASG_METRICS_COLLLECTION_TAG_NAME = "ASG_METRICS_COLLLECTION_ENABLED"
    initial_retry_delay = 10
    attempts = 0

    #print(event)

    if not event["detail"]["eventName"] == "CreateNodegroup":
        print("invalid event.")
        return -1

    clusterName = event["detail"]["requestParameters"]["name"]
    nodegroupName = event["detail"]["requestParameters"]["nodegroupName"]
    try:
        metricsCollectionEnabled = event["detail"]["requestParameters"]["tags"]
[ASG_METRICS_COLLLECTION_TAG_NAME]
    except KeyError:
        print(ASG_METRICS_COLLLECTION_TAG_NAME, "tag not found.")
        return

    # Check if metrics collection is enabled in tags
    if metricsCollectionEnabled.lower() != "true":
        print("Metrics collection is not enabled in nodegroup tags.")
        return

    # Get the name of the associated autoscaling group
    print("Getting the autoscaling group name for nodegroup=", nodegroupName, ",
 cluster=", clusterName )
    for i in range(0,10):
        try:
            autoScalingGroup =
 eks.describe_nodegroup(clusterName=clusterName,nodegroupName=nodegroupName)
["nodegroup"]["resources"]["autoScalingGroups"][0]["name"]
        except:
            attempts += 1
            print("Failed to obtain the associated autoscaling group for
 nodegroup", nodegroupName, "Retrying in", initial_retry_delay*attempts,
 "seconds.")
            time.sleep(initial_retry_delay*attempts)
        else:
```

```
            break

    print("Enabling metrics collection on autoscaling group ", autoScalingGroup)

    # Enable metrics collection in the autoscaling group
    try:
        enableMetricsCollection =
 autoscaling.enable_metrics_collection(AutoScalingGroupName=autoScalingGroup,Granularity="1
    except:
        print("Unable to enable metrics collection on nodegroup=",nodegroup)
    print("Enabled metrics collection on nodegroup", nodegroupName)
EOF
```

6. Créez un package de déploiement.

```
cd /tmp
zip function.zip lambda-handler.py
```

7. Créez une fonction Lambda.

```
LAMBDA_ARN=$(aws lambda create-function --function-name asg-enable-metrics-
collection \
  --zip-file fileb://function.zip --handler lambda-handler.lambda_handler \
  --runtime python3.9 \
  --timeout 600 \
  --role $LAMBDA_ROLE \
  --output text \
  --query 'FunctionArn')
echo $LAMBDA_ARN
```

8. Créez une EventBridge règle.

```
RULE_ARN=$(aws events put-rule --name CreateNodegroupRuleToLambda \
  --event-pattern "{\"source\":[\"aws.eks\"],\"detail-type\":[\"AWS API Call via
 CloudTrail\"],\"detail\":{\"eventName\":[\"CreateNodegroup\"],\"eventSource\":
[\"eks.amazonaws.com\"]}}" \
  --output text \
  --query 'RuleArn')
echo $RULE_ARN
```

9. Ajoutez la fonction Lambda en tant que cible.

```
aws events put-targets --rule CreateNodegroupRuleToLambda \
```

```
  --targets "Id"="1","Arn"="$LAMBDA_ARN"
```

10. Ajoutez une politique qui permet d' EventBridge invoquer la fonction Lambda.

```
aws lambda add-permission \
  --function-name asg-enable-metrics-collection \
  --statement-id CreateNodegroupRuleToLambda \
  --action 'lambda:InvokeFunction' \
  --principal events.amazonaws.com \
  --source-arn $RULE_ARN
```

La fonction Lambda permet la collecte des métriques de groupe Auto Scaling pour tous les groupes de nœuds gérés auxquels vous attribuez une balise `ASG_METRICS_COLLLECTION_ENABLED` définie sur `TRUE`. Pour vérifier que la collecte des métriques de groupe Auto Scaling est activée, accédez au groupe Auto Scaling associé sur la console Amazon EC2. Dans l'onglet Monitoring (Surveillance), la case Enable (Activer) doit être cochée.

Prise en charge du module complémentaire Amazon EKS pour l'opérateur ADOT

Amazon EKS prend en charge l'utilisation de la AWS Management Console, la AWS CLI et l'API Amazon EKS pour installer et gérer l'opérateur AWS Distro pour OpenTelemetry (ADOT). Il est ainsi plus facile de permettre à vos applications exécutées sur Amazon EKS d'envoyer des données métriques et de suivi à plusieurs options de service de surveillance comme Amazon CloudWatch, Prometheus et X-Ray.

Pour plus d'informations, consultez Démarrage avec AWS Distro for OpenTelemetry à l'aide des modules complémentaires EKS dans la documentation AWS Distro for OpenTelemetry.

Plus de AWS services intégrés à Amazon EKS

Outre les services décrits dans d'autres sections, Amazon EKS propose d'autres AWS services pour fournir des solutions supplémentaires. Cette rubrique identifie certains des autres services qui utilisent Amazon EKS pour ajouter des fonctionnalités, ou des services qu'Amazon EKS utilise pour effectuer des tâches.

Rubriques

Création de ressources Amazon EKS avec AWS CloudFormation

Amazon EKS est intégré à AWS CloudFormation, un service qui vous aide à modéliser et à configurer vos ressources AWS, afin d'accélérer la création et la gestion de vos ressources et de votre infrastructure. Vous créez un modèle qui décrit toutes les ressources AWS que vous souhaitez, par exemple un cluster Amazon EKS, et AWS CloudFormation approvisionne et configure ces ressources automatiquement.

Lorsque vous utilisez AWS CloudFormation, vous pouvez réutiliser votre modèle pour configurer vos ressources Amazon EKS de manière cohérente et répétée. Il suffit de décrire vos ressources une fois, puis d'allouer les mêmes ressources à l'infini dans plusieurs comptes et régions AWS.

Amazon EKS et modèles AWS CloudFormation

Pour allouer et configurer des ressources pour Amazon EKS et les services associés, vous devez comprendre les modèles AWS CloudFormation. Les modèles sont des fichiers texte formatés en JSON ou YAML. Ces modèles décrivent les ressources que vous souhaitez allouer dans vos

piles AWS CloudFormation. Si JSON ou YAML ne vous est pas familier, vous pouvez utiliser AWS CloudFormation Designer pour vous aider à démarrer avec des modèles AWS CloudFormation. Pour plus d'informations, consultez Qu'est-ce qu'AWS CloudFormation Designer ? dans le Guide de l'utilisateur AWS CloudFormation.

Amazon EKS prend en charge la création de clusters et de groupes de nœuds dans AWS CloudFormation. Pour plus d'informations, notamment des exemples de modèles JSON et YAML pour vos ressources Amazon EKS, consultez la référence au type de ressource Amazon EKS dans le Guide de l'utilisateur AWS CloudFormation.

En savoir plus sur AWS CloudFormation

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

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

Amazon EKS et AWS Local Zones

Une zone locale AWS est une extension d'une Région AWS située à proximité géographique de vos utilisateurs. Les Local Zones ont leurs propres connexions à Internet et prennent en charge AWS Direct Connect. Les ressources créées dans une zone locale peuvent servir les utilisateurs locaux avec des communications à faible latence. Pour plus d'informations, consultez Local Zones.

Amazon EKS prend en charge certaines ressources dans Local Zones. Cela inclut les nœuds Amazon EC2 autogérés, les volumes Amazon EBS et les Application Load Balancers (ALB). Nous vous recommandons de tenir compte des points suivants lorsque vous utilisez Local Zones dans le cadre de votre cluster Amazon EKS.

Nœuds

Vous ne pouvez pas créer de groupes de nœuds gérés ou des nœuds Fargate dans Local Zones avec Amazon EKS. Toutefois, vous pouvez créer des nœuds Amazon EC2 autogérés dans Local Zones à l'aide de l'API Amazon EC2, de AWS CloudFormation, ou d'`eksctl`. Pour de plus amples informations, veuillez consulter Nœuds autogérés.

Architecture réseau

- Le plan de contrôle Kubernetes géré par Amazon EKS s'exécute toujours dans la Région AWS. Le plan de contrôle Kubernetes géré par Amazon EKS ne peut pas s'exécuter dans la zone locale. Étant donné que les zones locales apparaissent comme un sous-réseau au sein de votre VPC, Kubernetes voit vos ressources de zone locale comme faisant partie de ce sous-réseau.
- Le cluster Kubernetes pour Amazon EKS communique avec les instances Amazon EC2 que vous exécutez dans la Région AWS ou la zone locale en utilisant les interfaces réseau Elastic gérées par Amazon EKS. Pour en savoir plus sur l'architecture réseau Amazon EKS, consultez Mise en réseau d'Amazon EKS.
- Contrairement aux sous-réseaux régionaux, Amazon EKS ne peut pas placer d'interfaces réseau dans vos sous-réseaux de Local Zone. Cela signifie que vous ne devez pas spécifier de sous-réseaux de Local Zone lorsque vous créez votre cluster.

Conteneurs Deep Learning

AWS Deep Learning Containers est un ensemble d'images Docker pour la formation et le service de modèles dans TensorFlow sur Amazon EKS et Amazon Elastic Container Service (Amazon ECS). Les conteneurs de deep learning fournissent des environnements optimisés avec des bibliothèques TensorFlow, NVIDIA CUDA (pour les instances GPU) et Intel MKL (pour les instances CPU), et sont disponibles dans Amazon ECR.

Pour démarrer avec les conteneurs de deep learning AWS sur Amazon EKS, consultez la rubrique Configuration d'Amazon EKS du Guide du développeur de conteneurs de deep learning AWS.

Amazon VPC Lattice

Amazon VPC Lattice est un service de mise en réseau d'applications entièrement géré intégré directement à l'infrastructure réseau AWS que vous pouvez utiliser pour connecter, sécuriser et surveiller vos services sur plusieurs comptes et clouds privés virtuels (VPC). Avec Amazon EKS, vous pouvez tirer parti d'Amazon VPC Lattice en utilisant le Contrôleur d'API Gateway AWS, une implémentation de l'API Gateway Kubernetes. À l'aide d'Amazon VPC Lattice, vous pouvez configurer la connectivité entre clusters avec une sémantique Kubernetes standard de manière simple et cohérente. Pour commencer à utiliser Amazon VPC Lattice avec Amazon EKS, consultez le Guide de l'utilisateur du contrôleur d'API Gateway AWS.

AWS Resilience Hub

AWS Resilience Hub évalue la résilience d'un cluster Amazon EKS en analysant son infrastructure. AWS Resilience Hub utilise la configuration de contrôle d'accès basé sur les rôles (RBAC, role-based access control) Kubernetes pour évaluer les charges de travail Kubernetes déployées sur votre cluster. Pour plus d'informations, veuillez consulter la rubrique Activation de l'accès AWS Resilience Hub à votre cluster Amazon EKS dans le Guide de l'utilisateur AWS Resilience Hub.

Détectez les menaces avec Amazon GuardDuty

Amazon GuardDuty est un service de détection des menaces qui vous aide à protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, vous surveillez GuardDuty en permanence les différentes sources de journaux et l'activité d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

Parmi les autres fonctionnalités, GuardDuty propose les deux fonctionnalités suivantes qui détectent les menaces potentielles pour vos clusters EKS : la protection EKS et la surveillance du temps d'exécution.

Protection EKS

Cette fonctionnalité fournit une couverture de détection des menaces pour vous aider à protéger les clusters Amazon EKS en surveillant les journaux Kubernetes d'audit associés. Kubernetesles journaux d'audit capturent les actions séquentielles au sein de votre cluster, notamment les activités des utilisateurs, des applications utilisant l'KubernetesAPI et du plan de contrôle. Par exemple, GuardDuty peut identifier que les API appelées pour potentiellement altérer les ressources d'un Kubernetes cluster ont été invoquées par un utilisateur non authentifié.

Lorsque vous activez EKS Protection, vous ne GuardDuty pourrez accéder à vos journaux d'audit Amazon EKS que pour une détection continue des menaces. S'il GuardDuty identifie une menace potentielle pour votre cluster, il génère une découverte du journal Kubernetes d'audit associée d'un type spécifique. Pour plus d'informations sur les types de résultats disponibles dans les journaux d'Kubernetesaudit, consultez la section Types de résultats des journaux Kubernetes d'audit dans le guide de GuardDuty l'utilisateur Amazon.

Pour plus d'informations, consultez EKS Protection dans le guide de GuardDuty l'utilisateur Amazon.

Surveillance d'exécution

Cette fonctionnalité surveille et analyse les événements au niveau du système d'exploitation, du réseau et des fichiers pour vous aider à détecter les menaces potentielles dans des AWS charges de travail spécifiques de votre environnement.

Lorsque vous activez la surveillance du temps d'exécution et que vous installez l' GuardDuty agent dans vos clusters Amazon EKS, GuardDuty commence à surveiller les événements d'exécution associés à ce cluster. S'il GuardDuty identifie une menace potentielle pour votre cluster, il génère un résultat de surveillance du temps d'exécution associé. Par exemple, une menace peut potentiellement commencer par compromettre un conteneur unique qui exécute une application Web vulnérable. Cette application Web peut disposer d'autorisations d'accès aux conteneurs et aux charges de travail sous-jacents. Dans ce scénario, des informations d'identification mal configurées peuvent potentiellement élargir l'accès au compte et aux données qui y sont stockées.

Pour configurer la surveillance du temps d'exécution, vous devez installer l' GuardDuty agent sur votre cluster en tant que module complémentaire Amazon EKS. Pour plus d'informations sur le module complémentaire, consultezModules complémentaires Amazon EKS disponibles sur Amazon EKS.

Pour plus d'informations, consultez la section Runtime Monitoring dans le guide de GuardDuty l'utilisateur Amazon.

Utilisation d'Amazon Security Lake avec Amazon EKS

Amazon Security Lake est un service de lac de données de sécurité entièrement géré qui vous permet de centraliser les données de sécurité provenant de diverses sources, notamment Amazon EKS. En intégrant Amazon EKS à Security Lake, vous pouvez obtenir des informations plus approfondies sur les activités effectuées sur vos Kubernetes ressources et améliorer le niveau de sécurité de vos clusters Amazon EKS.

> **Note**
>
> Pour plus d'informations sur l'utilisation de Security Lake avec Amazon EKS et sur la configuration des sources de données, consultez la documentation Amazon Security Lake.

Avantages de l'utilisation de Security Lake avec Amazon Amazon EKS

Données de sécurité centralisées : Security Lake collecte et centralise automatiquement les données de sécurité de vos clusters Amazon EKS, ainsi que les données provenant d'autres AWS services, de fournisseurs de SaaS, de sources sur site et de sources tierces. Cela fournit une vue complète de votre posture de sécurité dans l'ensemble de votre organisation.

Format de données standardisé — Security Lake convertit les données collectées au format Open Cybersecurity Schema Framework (OCSF), qui est un schéma open source standard. Cette normalisation facilite l'analyse et l'intégration avec d'autres outils et services de sécurité.

Détection des menaces améliorée : en analysant les données de sécurité centralisées, notamment les journaux du plan de contrôle Amazon EKS, vous pouvez détecter plus efficacement les activités potentiellement suspectes au sein de vos clusters Amazon EKS. Cela permet d'identifier les incidents de sécurité et d'y répondre rapidement.

Gestion simplifiée des données : Security Lake gère le cycle de vie de vos données de sécurité grâce à des paramètres de rétention et de réplication personnalisables. Cela simplifie les tâches de gestion des données et garantit que vous conservez les données nécessaires à des fins de conformité et d'audit.

Activation de Security Lake pour Amazon EKS

Pour commencer à utiliser Security Lake avec Amazon EKS, procédez comme suit :

1. Activez la journalisation du plan de contrôle Amazon EKS pour vos clusters EKS. Reportez-vous à la section Activation et désactivation des journaux du plan de contrôle pour obtenir des instructions détaillées.
2. Ajoutez les journaux d'audit Amazon EKS en tant que source dans Security Lake. Security Lake commencera alors à collecter des informations détaillées sur les activités effectuées sur les ressources Kubernetes exécutées dans vos clusters EKS.
3. Configurez les paramètres de rétention et de réplication de vos données de sécurité dans Security Lake en fonction de vos besoins.
4. Utilisez les données OCSF normalisées stockées dans Security Lake pour la réponse aux incidents, les analyses de sécurité et l'intégration avec d'autres AWS services ou outils tiers. Par exemple, vous pouvez générer des informations de sécurité à partir des données Amazon Security Lake à l'aide d'Amazon OpenSearch Ingestion.

Analyse des journaux EKS dans Security Lake

Security Lake normalise les événements du journal EKS au format OCSF, ce qui facilite l'analyse et la corrélation des données avec d'autres événements de sécurité. Vous pouvez utiliser différents outils et services, tels qu'Amazon Athena, Amazon QuickSight ou des outils d'analyse de sécurité tiers, pour interroger et visualiser les données normalisées.

Pour plus d'informations sur le mappage OCSF pour les événements du journal EKS, reportez-vous à la référence de mappage dans le référentiel OCSF GitHub .

Amazon Detective

Amazon Detective vous permet d'analyser, d'enquêter et d'identifier rapidement la cause racine des résultats de sécurité ou des activités suspectes. Detective collecte automatiquement les données du journal à partir de vos AWS ressources. Detective utilise ensuite le machine learning, l'analyse statistique et la théorie des graphiques pour générer des visualisations qui vous aideront à mener des investigations de sécurité plus rapides et plus efficaces. Les agrégations de données, les résumés et le contexte prédéfinis de Detective vous aident à analyser et à déterminer rapidement la nature et l'étendue des éventuels problèmes de sécurité. Pour plus d'informations, consultez le Guide de l'utilisateur Amazon Detective.

Detective organise Kubernetes et AWS collecte des données pour obtenir des résultats tels que :

- Les détails du cluster Amazon EKS, y compris l'identité IAM qui a créé le cluster et la fonction du service du cluster. Vous pouvez étudier l'activité AWS et l'activité des Kubernetes API de ces identités IAM avec Detective.
- Les détails du conteneur, tels que l'image et le contexte de sécurité. Vous pouvez également examiner les détails des Pods terminés.
- L'activité de l'API Kubernetes, y compris les tendances générales de l'activité de l'API et les détails sur les appels spécifiques de l'API. Par exemple, vous pouvez afficher le nombre d'appels API Kubernetes réussis et échoués qui ont été émis au cours d'une période sélectionnée. En outre, la section sur les appels d'API récemment observés peut être utile pour identifier une activité suspecte.

Les journaux d'audit Amazon EKS sont un package de source de données facultatif qui peut être ajouté à votre graphe de comportement Detective. Vous pouvez afficher les packages source facultatifs disponibles et leur état dans votre compte. Pour plus d'informations, veuillez consulter

la rubrique Journaux d'audit d'Amazon EKS pour Detective dans le Guide de l'utilisateur Amazon Detective.

Utilisation d'Amazon Detective avec Amazon EKS

Pour examiner les résultats d'un cluster Amazon EKS

Avant de pouvoir examiner les résultats, Detective doit être activé pendant au moins 48 heures au même endroit Région AWS que votre cluster. Pour plus d'informations, veuillez consulter la rubrique Configuration d'Amazon Detective dans le Guide de l'utilisateur Amazon Detective.

1. Ouvrez la console Detective à l'adresse suivante : https://console.aws.amazon.com/detective/.
2. Dans le volet de navigation de gauche, sélectionnez Search (Rechercher).
3. Sélectionnez Choose type (Choisir le type), puis sélectionnez EKS cluster.
4. Saisissez le nom du cluster ou l'ARN, puis choisissez Search (Rechercher).
5. Dans les résultats de la recherche, sélectionnez le nom du cluster dont vous souhaitez consulter l'activité. Pour plus d'informations sur ce que vous pouvez consulter, consultez la rubrique Activité globale de l'API Kubernetes impliquant un cluster Amazon EKS dans le Guide de l'utilisateur Amazon Detective.

Dépannage d'Amazon EKS

Ce chapitre traite de certaines erreurs courantes que vous pouvez rencontrer lorsque vous utilisez Amazon EKS, ainsi que des solutions. Si vous avez besoin de dépanner des zones spécifiques d'Amazon EKS, consultez les rubriques Dépannage IAM, Résolution des problèmes dans Amazon EKS Connector et Dépannage d'ADOT à l'aide de modules complémentaires EKS.

Pour obtenir d'autres informations permettant de résoudre les problèmes, consultez la section Contenu du centre de connaissances concernant Amazon Elastic Kubernetes Service sur AWS re:Post.

Capacité insuffisante

Si vous recevez le message d'erreur suivant lorsque vous tentez de créer un cluster Amazon EKS, cela signifie que l'une des zones de disponibilité spécifiées ne dispose pas d'une capacité suffisante pour prendre en charge un cluster.

```
Cannot create cluster 'example-cluster' because region-1d, the targeted Availability Zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these Availability Zones: region-1a, region-1b, region-1c
```

Essayez à nouveau de créer votre cluster avec des sous-réseaux de votre VPC de cluster hébergés dans les zones de disponibilité renvoyés par ce message d'erreur.

Il existe des zones de disponibilité dans lesquelles un cluster ne peut pas résider. Comparez les zones de disponibilité dans lesquelles se trouvent vos sous-réseaux avec la liste des zones de disponibilité figurant dans les Exigences et considérations requises pour les sous-réseaux.

Les nœuds ne parviennent pas à joindre le cluster

Quelques raisons courantes empêchent les nœuds de joindre le cluster :

- Si les nœuds sont des nœuds gérés, Amazon EKS ajoute des entrées à la `ConfigMap` `aws-auth` lorsque vous créez le groupe de nœuds. Si l'entrée a été supprimée ou modifiée, vous devez l'ajouter de nouveau. Pour plus d'informations, entrez **`eksctl create iamidentitymapping`**

`--help` dans votre terminal. Vous pouvez afficher vos entrées actuelles de `ConfigMap aws-auth` en remplaçant *`my-cluster`* dans la commande suivante par le nom de votre cluster et en exécutant la commande modifiée : **`eksctl get iamidentitymapping --cluster my-cluster`**. L'ARN du rôle que vous spécifiez ne peut pas inclure de chemin autre que `/`. Par exemple, si le nom de votre rôle est `development/apps/my-role`, vous devez le remplacer par `my-role` lorsque vous spécifiez l'ARN du rôle. Assurez-vous que vous spécifiez l'ARN pour le rôle IAM du nœud (et non l'ARN du profil d'instance).

Si les nœuds sont autogérés et que vous n'avez pas créé d'entrées d'accès pour l'ARN du rôle IAM du nœud, exécutez les mêmes commandes répertoriées pour les nœuds gérés. Si vous avez créé une entrée d'accès pour l'ARN du rôle IAM de votre nœud, il se peut qu'elle ne soit pas correctement configurée dans l'entrée d'accès. Assurez-vous que l'ARN du rôle IAM du nœud (et non l'ARN du profil d'instance) est spécifié comme ARN principal dans votre entrée de `ConfigMap aws-auth` ou dans votre entrée d'accès. Pour plus d'informations sur les entrées d'accès, consultez Gérer les entrées d'accès.

- Le AWS CloudFormation modèle ClusterNamede votre nœud ne correspond pas exactement au nom du cluster que vous souhaitez que vos nœuds rejoignent. Une valeur incorrecte pour ce champ génère une configuration incorrecte du fichier `/var/lib/kubelet/kubeconfig` du nœud, et les nœuds ne seront pas en mesure de joindre le cluster.
- Le nœud n'est pas étiqueté comme appartenant au cluster. La balise suivante doit être appliquée à vos nœuds, où *`my-cluster`* est remplacé par le nom de votre cluster.

Clé	Valeur
`kubernetes.io/cluster/` *`my-cluster`*	`owned`

- Les nœuds peuvent ne pas être en mesure d'accéder au cluster à l'aide d'une adresse IP publique. Assurez-vous qu'une adresse IP publique est attribuée aux nœuds déployés dans les sous-réseaux publics. Si ce n'est pas le cas, vous pouvez associer une adresse IP élastique à un nœud après son lancement. Pour plus d'informations, consultez Association d'une adresse IP élastique à une instance en cours d'exécution ou à une interface réseau. Si le sous-réseau public n'est pas défini pour attribuer automatiquement des adresses IP publiques aux instances qui y sont déployées, nous vous recommandons d'activer ce paramètre. Pour plus d'informations, consultez Modification de l'attribut d'adressage IPv4 public de votre sous-réseau. Si le nœud est déployé sur un sous-réseau privé, ce sous-réseau doit disposer d'une route vers une passerelle NAT à laquelle une adresse IP publique est attribuée.

- Le AWS STS point de terminaison sur Région AWS lequel vous déployez les nœuds n'est pas activé pour votre compte. Pour activer la région, voir Activation et désactivation AWS STS dans un Région AWS.
- Le nœud n'a pas d'entrée DNS privée, ce qui fait que le journal `kubelet` contient une erreur `node "" not found`. Assurez-vous que le VPC sur lequel le nœud est créé a des valeurs définies pour `domain-name` et `domain-name-servers` comme `Options` dans un `DHCP options set`. Les valeurs par défaut sont `domain-name:<region>.compute.internal` et `domain-name-servers:AmazonProvidedDNS`. Pour en savoir plus, consultez Jeux d'options DHCP dans le Guide de l'utilisateur Amazon VPC.
- Si les nœuds du groupe de nœuds gérés ne se connectent pas au cluster dans les 15 minutes, un problème de santé « NodeCreation Défaillance » sera émis et l'état de la console sera défini sur`Create failed`. Pour les Windows AMI dont les temps de lancement sont lents, ce problème peut être résolu grâce au lancement rapide.

Pour identifier et résoudre les problèmes courants qui empêchent les composants master de rejoindre un cluster, vous pouvez utiliser le runbook`AWSSupport-TroubleshootEKSWorkerNode`. Pour plus d'informations, consultez `AWSSupport-TroubleshootEKSWorkerNode` dans la référence AWS Systems Manager Automation runbook.

Accès non autorisé ou refusé (**`kubectl`**)

Si vous obtenez l'une des erreurs suivantes lors de l'exécution de commandes `kubectl`, votre `kubectl` n'est pas correctement configuré pour Amazon EKS, ou les informations d'identification du principal IAM (rôle ou utilisateur) que vous utilisez ne sont pas mappées vers un nom d'utilisateur Kubernetes disposant des autorisations suffisantes pour des objets Kubernetes dans votre cluster Amazon EKS.

- `could not get token: AccessDenied: Access denied`
- `error: You must be logged in to the server (Unauthorized)`
- `error: the server doesn't have a resource type "svc"`

Cela peut être dû à l'une des raisons suivantes :

- Le cluster a été créé avec les informations d'identification d'un principal IAM et `kubectl` est configuré pour utiliser les informations d'identification d'un autre principal IAM. Pour résoudre ce problème, mettez à jour votre fichier `kube config` afin d'utiliser les informations d'identification à

l'origine de la création du cluster. Pour plus d'informations, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

- Si votre cluster répond aux exigences de plateforme minimales indiquées dans la section des conditions préalables des Gérer les entrées d'accès, aucune entrée d'accès n'existe auprès de votre principal IAM. Si elle existe, elle n'a pas les noms de groupe Kubernetes nécessaires définis ou n'a pas la bonne stratégie d'accès associée. Pour plus d'informations, consultez Gérer les entrées d'accès.
- Si votre cluster ne répond pas aux exigences de plateforme minimales indiquées dans la section Gérer les entrées d'accès, aucune entrée d'accès n'existe auprès de votre principal IAM dans la `ConfigMap` `aws-auth`. Si elle existe, elle n'est pas mappée vers les noms de groupes Kubernetes liés à un `Role` ou `ClusterRole` Kubernetes dotés des autorisations nécessaires. Pour plus d'informations sur les objets de contrôle d'autorisation basé sur les rôles (RBAC) de Kubernetes, consultez la rubrique Utilisation de l'autorisation RBAC de la documentation Kubernetes. Vous pouvez afficher vos entrées actuelles de `ConfigMap` `aws-auth` en remplaçant *`my-cluster`* dans la commande suivante par le nom de votre cluster et en exécutant la commande modifiée : **`eksctl get iamidentitymapping --cluster my-cluster`**. Si aucune entrée contenant l'ARN de votre principal IAM ne figure dans `ConfigMap`, entrez **`eksctl create iamidentitymapping --help`** dans votre terminal pour savoir comment en créer une.

Si vous installez et configurez le AWS CLI, vous pouvez configurer les informations d'identification IAM que vous utilisez. Pour plus d'informations, veuillez consulter configuration de l'outil AWS CLI dans le guide de l'utilisateur de l'outil AWS Command Line Interface . Vous pouvez également configurer `kubectl` pour utiliser un rôle IAM, si vous assumez un rôle IAM pour accéder aux objets Kubernetes de votre cluster. Pour plus d'informations, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

`hostname doesn't match`

La version Python de votre système doit être la version `2.7.9` ou une version ultérieure. Dans le cas contraire, vous recevrez `hostname doesn't match` des erreurs lors AWS CLI des appels vers Amazon EKS. Pour de plus amples informations, veuillez consulter Que sont les erreurs « hostname doesn't match » ? dans les Questions fréquentes (FAQ) relatives aux demandes Python.

getsockopt: no route to host

Docker s'exécute dans la plage d'adresses CIDR `172.17.0.0/16` dans les clusters Amazon EKS. Nous vous recommandons de veiller à ce que les sous-réseaux VPC de votre cluster ne chevauchent pas cette plage. Sinon, vous recevrez l'erreur suivante :

```
Error: : error upgrading connection: error dialing backend: dial tcp
 172.17.<nn>.<nn>:10250: getsockopt: no route to host
```

Instances failed to join the Kubernetes cluster

Si le message d'erreur s'affiche `Instances failed to join the Kubernetes cluster` dans le AWS Management Console, assurez-vous que l'accès au point de terminaison privé du cluster est activé ou que vous avez correctement configuré les blocs CIDR pour l'accès au point de terminaison public. Pour plus d'informations, consultez Contrôle d'accès au point de terminaison du cluster Amazon EKS..

Codes d'erreurs liées aux groupes de nœuds gérés

Si votre groupe de nœuds gérés rencontre un problème d'intégrité matérielle, Amazon EKS renvoie un code d'erreur pour vous aider à diagnostiquer le problème. Ces surveillances de l'état ne détectent pas les problèmes logiciels, car elles sont basées sur les surveillances de l'état d'Amazon EC2. La liste suivante décrit les codes d'erreur.

AccessDenied

> Amazon EKS ou un ou plusieurs de vos nœuds gérés ne parviennent pas à procéder à l'authentification ou à obtenir l'autorisation avec le serveur d'API de votre cluster Kubernetes. Pour de plus amples informations sur la résolution d'une cause courante, consultez Correction d'une cause courante d'erreurs `AccessDenied` pour les groupes de nœuds gérés. Les AMI Windows privées peuvent également provoquer ce code d'erreur à côté du message d'erreur `Not authorized for images`. Pour plus d'informations, consultez `Not authorized for images`.

AmiIdNotFound

> Nous n'avons pas pu trouver l'ID d'AMI associé à votre modèle de lancement. Assurez-vous que l'AMI existe et est partagée avec votre compte.

AutoScalingGroupNotTrouvé

Nous n'avons pas pu trouver le groupe Auto Scaling associé au groupe de nœuds gérés. Vous pouvez peut-être recréer un groupe Auto Scaling avec les mêmes paramètres pour effectuer une récupération.

ClusterUnreachable

Amazon EKS ou un ou plusieurs de vos nœuds gérés ne peuvent pas communiquer avec le serveur d'API de votre cluster Kubernetes. Cela peut se produire s'il y a des interruptions de réseau ou si les serveurs d'API temporisent le traitement des demandes.

Eco 2 SecurityGroup NotFound

Nous n'avons pas pu trouver le groupe de sécurité de cluster pour le cluster. Vous devez recréer votre cluster.

Eco 2 SecurityGroup DeletionFailure

Nous n'avons pas pu supprimer le groupe de sécurité d'accès à distance pour votre groupe de nœuds gérés. Supprimez toutes les dépendances du groupe de sécurité.

Eco 2 LaunchTemplate NotFound

Nous n'avons pas pu trouver le modèle de lancement Amazon EC2 pour votre groupe de nœuds gérés. Vous devez recréer votre groupe de nœuds pour effectuer une récupération.

Eco 2 LaunchTemplate VersionMismatch

La version du modèle de lancement Amazon EC2 de votre groupe de nœuds gérés ne correspond pas à la version créée par Amazon EKS. Vous pouvez peut-être revenir à la version Amazon EKS créée pour effectuer une récupération.

IamInstanceProfileNotTrouvé

Nous n'avons pas trouvé le profil d'instance IAM pour votre groupe de nœuds gérés. Vous pouvez peut-être recréer un profil d'instance avec les mêmes paramètres pour effectuer une récupération.

IamNodeRoleNotTrouvé

Nous n'avons pas pu trouver le rôle IAM pour votre groupe de nœuds gérés. Vous pouvez peut-être recréer un rôle IAM avec les mêmes paramètres pour effectuer une récupération.

AsgInstanceLaunchFailures

Votre groupe Auto Scaling rencontre des problèmes lors d'une tentative de lancement d'instances.

NodeCreationDéfaillance

Vos instances lancées ne peuvent pas s'enregistrer auprès de votre cluster Amazon EKS. Les causes courantes de cet échec sont des autorisations de rôle IAM de nœud insuffisantes ou l'absence d'un accès Internet sortant pour les nœuds. Vos nœuds doivent répondre à l'une des exigences suivantes :

- Accès à Internet à l'aide d'une adresse IP publique. Le groupe de sécurité associé au sous-réseau dans lequel se trouve le nœud doit autoriser la communication. Pour plus d'informations, consultez Exigences et considérations requises pour les sous-réseaux et Considérations et exigences relatives aux groupes de sécurité Amazon EKS.
- Vos nœuds et VPC doivent répondre aux exigences dans Exigences relatives aux clusters privés.

InstanceLimitDépassé

Votre AWS compte n'est plus en mesure de lancer d'autres instances du type d'instance spécifié. Vous pouvez demander une augmentation de la limite d'instances Amazon EC2 pour effectuer une récupération.

InsufficientFreeAdresses

Un ou plusieurs sous-réseaux associés à votre groupe de nœuds gérés ne disposent pas d'un nombre suffisant d'adresses IP pour de nouveaux nœuds.

InternalFailure

Ces erreurs sont généralement dues à un problème côté serveur Amazon EKS.

Correction d'une cause courante d'erreurs **`AccessDenied`** pour les groupes de nœuds gérés

La cause la plus courante d'erreurs `AccessDenied` lors de la réalisation d'opérations sur des groupes de nœuds gérés est l'absence de `eks:node-manager` `ClusterRole` ou `ClusterRoleBinding`. Amazon EKS configure ces ressources dans votre cluster dans le cadre de l'onboarding avec les groupes de nœuds gérés, et celles-ci sont nécessaires pour gérer les groupes de nœuds.

La `ClusterRole` peut changer au fil du temps, mais elle doit ressembler à l'exemple suivant :

```
apiVersion: rbac.authorization.k8s.io/v1
```

```
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
```

La `ClusterRoleBinding` peut changer au fil du temps, mais elle doit ressembler à l'exemple suivant :

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
```

```
  kind: User
  name: eks:node-manager
```

Vérifiez que la `eks:node-manager` `ClusterRole` existe.

```
kubectl describe clusterrole eks:node-manager
```

Si elle est présente, comparez la sortie à l'exemple de la `ClusterRole`précédente.

Vérifiez que la `eks:node-manager` `ClusterRoleBinding` existe.

```
kubectl describe clusterrolebinding eks:node-manager
```

Si elle est présente, comparez la sortie à l'exemple de la `ClusterRoleBinding`précédente.

Si vous avez identifié une `ClusterRole` ou `ClusterRoleBinding` manquante ou cassée comme la cause d'une erreur `AcessDenied` lors de la demande d'opérations de groupe de nœuds gérés, vous pouvez les restaurer. Enregistrez le contenu suivant dans un fichier nommé *`eks-node-manager-role.yaml`*.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
```

```
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

Appliquez le fichier.

```
kubectl apply -f eks-node-manager-role.yaml
```

Réessayez l'opération de groupe de nœuds pour voir si cela a résolu votre problème.

Not authorized for images

L'une des causes potentielles d'un message d'erreur `Not authorized for images` est l'utilisation d'une AMI Windows Amazon EKS privée pour lancer des groupes de nœuds Windows gérés. Après avoir publié de nouvelles Windows AMI, AWS les AMI datant de plus de 4 mois deviennent privées, ce qui les rend inaccessibles. Si votre groupe de nœuds gérés utilise une Windows AMI privée, pensez à mettre à jour votre groupe de nœuds Windows gérés. Bien que nous ne puissions pas garantir l'accès aux AMI qui ont été rendues privées, vous pouvez demander l'accès en déposant un ticket auprès du AWS Support. Pour plus d'informations, consultez la section Correctifs, mises à jour de sécurité et identifiants AMI dans le guide de l'utilisateur Amazon EC2.

Le nœud est en **NotReady** état

Si votre nœud entre dans un `NotReady` état, cela indique probablement qu'il n'est pas en bon état et qu'il n'est pas disponible pour en planifier un nouveauPods. Cela peut se produire pour diverses raisons, telles que le fait que le nœud ne dispose pas de ressources suffisantes pour le processeur, la mémoire ou l'espace disque disponible.

Pour les Windows AMI optimisées pour Amazon EKS, aucune réservation n'est prévue pour les ressources de calcul spécifiées par défaut dans la `kubelet` configuration. Pour éviter les problèmes de ressources, vous pouvez réserver des ressources de calcul pour les processus du système en fournissant `kubelet` des valeurs de configuration pour `kube-reserved`et/ou `system-reserved`. Pour ce faire, utilisez le paramètre de `-KubeletExtraArgs` ligne de commande du script bootstrap. Pour plus d'informations, consultez la section Reserve Compute Resources for System Daemons dans la Kubernetes documentation et les paramètres de configuration du script Bootstrap dans ce guide de l'utilisateur.

Outil de collecte de journaux CNI

Le Amazon VPC CNI plugin for pour Kubernetes dispose de son propre script de dépannage disponible sur les nœuds au niveau de `/opt/cni/bin/aws-cni-support.sh`. Vous pouvez utiliser le script pour collecter des journaux de diagnostic pour les cas de support et le dépannage général.

Utilisez la commande suivante pour exécuter le script sur votre nœud :

```
sudo bash /opt/cni/bin/aws-cni-support.sh
```

> **Note**
>
> Si le script n'est pas présent dans cet emplacement, alors le conteneur CNI n'a pas pu être exécuté. Vous pouvez manuellement télécharger et exécuter le script à l'aide de la commande suivante :
>
> ```
> curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/log-collector-script/linux/eks-log-collector.sh
> sudo bash eks-log-collector.sh
> ```

Le script collecte les informations de diagnostic suivantes. La version CNI que vous avez déployée peut être antérieure à la version du script.

```
      This is version 0.6.1. New versions can be found at https://github.com/awslabs/
amazon-eks-ami

Trying to collect common operating system logs...
Trying to collect kernel logs...
Trying to collect mount points and volume information...
Trying to collect SELinux status...
Trying to collect iptables information...
Trying to collect installed packages...
Trying to collect active system services...
Trying to collect Docker daemon information...
Trying to collect kubelet information...
Trying to collect L-IPAMD information...
Trying to collect sysctls information...
Trying to collect networking information...
Trying to collect CNI configuration information...
Trying to collect running Docker containers and gather container data...
Trying to collect Docker daemon logs...
Trying to archive gathered information...

 Done... your bundled logs are located in /var/
log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

Les informations de diagnostic sont collectées et stockés dans :

```
/var/log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

Le réseau d'exécution du conteneur n'est pas prêt

Vous pouvez recevoir une erreur `Container runtime network not ready` et des erreurs d'autorisation similaires aux suivantes :

```
4191 kubelet.go:2130] Container runtime network not ready: NetworkReady=false
 reason:NetworkPluginNotReady message:docker: network plugin is not ready: cni config
 uninitialized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

```
4191 kubelet_node_status.go:106] Unable to register node
 "ip-10-40-175-122.ec2.internal" with API server: Unauthorized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

Les raisons possibles sont les suivantes :

1. Soit vous n'avez pas de `ConfigMap aws-auth` sur votre cluster, soit celui-ci n'inclut pas d'entrées pour le rôle IAM avec lequel vous avez configuré vos nœuds.

   Cette entrée `ConfigMap` est nécessaire si vos nœuds répondent à l'un des critères suivants :

   - Nœuds gérés dans un cluster avec n'importe quelle version Kubernetes ou version de plateforme.
   - Nœuds autogérés dans un cluster antérieur à l'une des versions de plateforme répertoriées dans la section des conditions préalables de la rubrique Gérer les entrées d'accès.

   Pour résoudre le problème, affichez les entrées existantes de votre `ConfigMap` en remplaçant *`my-cluster`* dans la commande suivante par le nom de votre cluster et en exécutant la commande modifiée : **`eksctl get iamidentitymapping --cluster my-cluster`**. Si vous commande renvoie un message d'erreur, cela peut être dû au fait que votre cluster ne possède pas de `ConfigMap aws-auth`. La commande suivante ajoute une entrée à la `ConfigMap`. Si la `ConfigMap` n'existe pas, la commande peut également la créer. Remplacez *`111122223333`* par l' Compte AWS ID du rôle IAM et *`MyAmazoneks NodeRole`* par le nom du rôle de votre nœud.

   ```
   eksctl create iamidentitymapping --cluster my-cluster \
       --arn arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --group
    system:bootstrappers,system:nodes \
       --username system:node:{{EC2PrivateDNSName}}
   ```

   L'ARN du rôle que vous spécifiez ne peut pas inclure de chemin autre que `/`. Par exemple, si le nom de votre rôle est `development/apps/my-role`, vous devez le remplacer par `my-role` lorsque vous spécifiez l'ARN du rôle. Assurez-vous que vous spécifiez l'ARN pour le rôle IAM du nœud (et non l'ARN du profil d'instance).

2. Vos nœuds autogérés font partie d'un cluster dont la version de plateforme est la version minimale répertoriée dans les conditions requises de la rubrique Gérer les entrées d'accès, mais aucune entrée n'est répertoriée dans la `ConfigMap aws-auth` (voir point précédent) pour le rôle IAM du nœud ou aucune entrée d'accès n'existe pour le rôle. Pour résoudre le problème, affichez vos entrées existantes en remplaçant *`my-cluster`* dans la commande suivante par le nom de

votre cluster et en exécutant la commande modifiée : **aws eks list-access-entries --cluster-name *my-cluster***. La commande suivante ajoute une entrée d'accès pour le rôle IAM du nœud. Remplacez *111122223333* par l' Compte AWS ID du rôle IAM et *MyAmazoneksNodeRole* par le nom du rôle de votre nœud. Si vous avez un nœud Windows, remplacez *EC2_Linux* par. **EC2_Windows** Assurez-vous que vous spécifiez l'ARN pour le rôle IAM du nœud (et non l'ARN du profil d'instance).

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --type EC2_Linux
```

Délai d'expiration de la liaison TLS

Lorsqu'un nœud est incapable d'établir une connexion avec le point de terminaison du serveur d'API public, vous pouvez voir une erreur similaire à l'erreur suivante.

```
server.go:233] failed to run Kubelet: could not init cloud provider "aws": error
 finding instance i-1111f2222f333e44c: "error listing AWS instances: \"RequestError:
 send request failed\\ncaused by: Post  net/http: TLS handshake timeout\""
```

Le processus `kubelet` se reproduira continuellement et testera le point de terminaison du serveur d'API. L'erreur peut également se produire temporairement au cours de toute procédure qui effectue une mise à jour continue du cluster dans le plan de contrôle, telle qu'une modification de configuration ou une mise à jour de version.

Pour résoudre le problème, vérifiez la table de routage et les groupes de sécurité pour vous assurer que le trafic provenant des nœuds peut atteindre le point de terminaison public.

InvalidClientTokenId

Si vous utilisez des rôles IAM pour les comptes de service d'un cluster en Chine Pod ou que vous l'avez `DaemonSet` déployé sur un cluster Région AWS, et que vous n'avez pas défini la variable d'`AWS_DEFAULT_REGION`environnement dans la spécification, le Pod ou `DaemonSet` peut recevoir le message d'erreur suivant :

```
An error occurred (InvalidClientTokenId) when calling the GetCallerIdentity operation:
 The security token included in the request is invalid
```

Pour résoudre le problème, vous devez ajouter la variable d'environnement `AWS_DEFAULT_REGION` à votre spécification de Pod ou de `DaemonSet`, comme indiqué dans l'exemple suivant de spécification de Pod.

```
apiVersion: v1
kind: Pod
metadata:
  name: envar-demo
  labels:
    purpose: demonstrate-envars
spec:
  containers:
  - name: envar-demo-container
    image: gcr.io/google-samples/node-hello:1.0
    env:
    - name: AWS_DEFAULT_REGION
      value: "region-code"
```

Expiration du certificat webhook d'admission VPC

Si le certificat utilisé pour signer le webhook d'admission VPC expire, l'état des nouveaux déploiements de Pod Windowsreste à `ContainerCreating`.

Pour résoudre le problème si vous disposez d'un support Windows hérité sur votre plan de données, consultez Renouvellement du certificat du webhook d'admission VPC. Si la version de votre cluster et de votre plateforme est postérieure à une version répertoriée dans les prérequis à la prise en charge de Windows, nous vous recommandons de supprimer la prise en charge héritée de Windows sur votre plan de données et de l'activer pour votre plan de contrôle. Une fois que vous l'avez fait, vous n'avez pas besoin de gérer le certificat Webhook. Pour plus d'informations, consultez Activation de la prise en charge de Windows pour votre cluster Amazon EKS.

Les groupes de nœuds doivent correspondre à la version de Kubernetes avant de mettre à niveau le plan de contrôle

Avant de mettre à niveau un plan de contrôle vers une nouvelle version de Kubernetes, la version mineure des nœuds gérés et Fargate de votre cluster doit être la même que la version actuelle de votre plan de contrôle. L'API `update-cluster-version` Amazon EKS rejette les demandes tant que vous n'avez pas mis à niveau tous les nœuds gérés par Amazon EKS vers la version actuelle du

cluster. Amazon EKS fournit des API pour mettre à niveau les nœuds gérés. Pour plus d'informations sur la mise à niveau de la version Kubernetes d'un groupe de nœuds gérés, consultez Mise à jour d'un groupe de nœuds gérés. Pour mettre à niveau la version d'un nœud Fargate, supprimez le pod représenté par le nœud et redéployez le pod après avoir mis à niveau votre plan de contrôle. Pour plus d'informations, consultez Mise à jour d'une version Kubernetes de cluster Amazon EKS.

Lors du lancement de nombreux nœuds, des erreurs `Too Many Requests` se produisent

Si vous lancez plusieurs nœuds simultanément, vous pouvez voir un message d'erreur dans les journaux d'exécution des données utilisateur Amazon EC2 indiquant `Too Many Requests`. Cela peut se produire parce que le plan de contrôle est surchargé d'appels `describeCluster`. Cette surcharge se traduit par une limitation, des nœuds qui ne parviennent pas à exécuter le script d'amorçage et des nœuds qui ne parviennent pas à rejoindre le cluster.

Assurez-vous que les arguments `--apiserver-endpoint`, `--b64-cluster-ca` et `--dns-cluster-ip` sont transmis au script d'amorçage du nœud. Lorsque vous incluez ces arguments, le script d'amorçage n'a pas besoin d'effectuer un appel `describeCluster`, ce qui permet d'éviter la surcharge du plan de contrôle. Pour plus d'informations, consultez Fournir des données utilisateur pour passer des arguments au fichier `bootstrap.sh` inclus avec une AMI Linux/Bottlerocket optimisée pour Amazon EKS.

Réponse d'erreur non autorisée HTTP 401 sur les requêtes du serveur API Kubernetes

Ces erreurs s'affichent si le jeton de compte de service d'un Pod a expiré sur un cluster.

Le serveur d'API Kubernetes de votre cluster Amazon EKS rejette les demandes dont les jetons datent de 90 jours. Dans les versions précédentes de Kubernetes, les jetons n'avaient pas de période d'expiration. Cela signifie que les clients qui s'appuient sur ces jetons doivent les actualiser dans l'heure. Pour empêcher le serveur d'API Kubernetes de rejeter votre demande en raison d'un jeton non valide, la version du SDK client Kubernetes utilisée par votre charge de travail doit être identique ou ultérieure aux versions suivantes :

- Go version `0.15.7` et ultérieure
- Python version `12.0.0` et ultérieure
- Java version `9.0.0` et ultérieure

- JavaScript version `0.10.3` et versions ultérieures
- Branche `master` Ruby
- Haskell version `0.3.0.0`
- C# version `7.0.5` et ultérieure

Vous pouvez identifier tous les Pods existants de votre cluster qui utilisent des jetons obsolètes. Pour plus d'informations, consultez comptes de service Kubernetes.

La version de la plateforme Amazon EKS est inférieure de plus de deux versions à la version actuelle de la plateforme

Cela peut se produire lorsqu'Amazon EKS n'est pas en mesure de mettre à jour automatiquement la version de la plateforme de votre cluster. Bien qu'il existe de nombreuses causes à cela, certaines des causes les plus courantes suivent. Si l'un de ces problèmes s'applique à votre cluster, il est possible qu'il fonctionne toujours. Sa version de plateforme ne sera tout simplement pas mise à jour par Amazon EKS.

Problème

Le Rôle IAM du cluster a été supprimé : ce rôle a été spécifié lors de la création du cluster. Vous pouvez voir quel rôle a été spécifié à l'aide de la commande suivante. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-cluster --name my-cluster --query cluster.roleArn --output text | cut
 -d / -f 2
```

L'exemple qui suit illustre un résultat.

```
eksClusterRole
```

Solution

Création d'un nouveau rôle IAM du cluster portant le même nom.

Problème

Un sous-réseau spécifié lors de la création du cluster a été supprimé : les sous-réseaux à utiliser avec le cluster ont été spécifiés lors de la création du cluster. Vous pouvez voir quels sous-réseaux

ont été spécifiés à l'aide de la commande suivante. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-cluster --name my-cluster --query cluster.resourcesVpcConfig.subnetIds
```

L'exemple qui suit illustre un résultat.

```
[
"subnet-EXAMPLE1",
"subnet-EXAMPLE2"
]
```

Solution

Vérifiez si les ID de sous-réseau existent dans votre compte.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" --query
 "Subnets[*].SubnetId"
```

L'exemple qui suit illustre un résultat.

```
[
"subnet-EXAMPLE3",
"subnet-EXAMPLE4"
]
```

Si les ID de sous-réseau renvoyés dans la sortie ne correspondent pas aux ID de sous-réseau spécifiés lors de la création du cluster, vous devez modifier les sous-réseaux utilisés par le cluster si vous souhaitez qu'Amazon EKS mette à jour le cluster. En effet, si vous avez spécifié plus de deux sous-réseaux lors de la création de votre cluster, Amazon EKS sélectionne de manière aléatoire les sous-réseaux que vous avez spécifiés pour y créer de nouvelles interfaces réseau Elastic. Ces interfaces réseau permettent au plan de contrôle de communiquer avec vos nœuds. Amazon EKS ne met pas à jour le cluster si le sous-réseau qu'il sélectionne n'existe pas. Vous n'avez aucun contrôle sur les sous-réseaux que vous avez spécifiés lors de la création du cluster dans lesquels Amazon EKS choisit de créer une nouvelle interface réseau.

Lorsque vous lancez une mise à jour de la version Kubernetes pour votre cluster, la mise à jour peut échouer pour la même raison.

Problème

Un groupe de sécurité spécifié lors de la création du cluster a été supprimé : si vous avez spécifié des groupes de sécurité lors de la création du cluster, vous pouvez voir leurs ID à l'aide de la commande suivante. Remplacez *my-cluster* par le nom de votre cluster.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.securityGroupIds
```

L'exemple qui suit illustre un résultat.

```
[
    "sg-EXAMPLE1"
]
```

Si `[]` est renvoyé, aucun groupe de sécurité n'a été spécifié lors de la création du cluster et un groupe de sécurité manquant n'est pas le problème. Si des groupes de sécurité sont renvoyés, vérifiez qu'ils existent dans votre compte.

Solution

Vérifiez si ces groupes de sécurité existent dans votre compte.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=$vpc_id" --query
 "SecurityGroups[*].GroupId"
```

L'exemple qui suit illustre un résultat.

```
[
"sg-EXAMPLE2"
]
```

Si les ID des groupes de sécurité renvoyés dans la sortie ne correspondent pas aux ID des groupes de sécurité spécifiés lors de la création du cluster, vous devez modifier les groupes de sécurité utilisés par le cluster si vous souhaitez qu'Amazon EKS mette à jour le cluster. Amazon EKS ne met pas à jour un cluster si les ID de groupe de sécurité spécifiés lors de la création du cluster n'existent pas.

Lorsque vous lancez une mise à jour de la version Kubernetes pour votre cluster, la mise à jour peut échouer pour la même raison.

Autres raisons pour lesquelles Amazon EKS ne met pas à jour la version de plateforme de votre cluster

- Vous n'avez pas au moins six (bien que nous en recommandions 16) adresses IP disponibles dans chacun des sous-réseaux que vous avez spécifiés lors de la création de votre cluster. Si vous n'avez pas assez d'adresses IP disponibles dans le sous-réseau, vous devez soit libérer des adresses IP dans le sous-réseau, soit modifier les sous-réseaux utilisés par le cluster pour utiliser des sous-réseaux ayant suffisamment d'adresses IP disponibles.
- Vous avez activé le chiffrement des secrets lorsque vous avez créé votre cluster et la AWS KMS clé que vous avez spécifiée a été supprimée. Si vous souhaitez qu'Amazon EKS mette à jour le cluster, vous devez créer un nouveau cluster

FAQ sur l'état du cluster et codes d'erreur avec chemins de résolution

Amazon EKS détecte les problèmes liés à vos clusters EKS et à l'infrastructure du cluster et les enregistre dans l'état du cluster. Vous pouvez détecter et résoudre les problèmes de cluster plus rapidement à l'aide des informations relatives à l'état du cluster. Cela vous permet de créer des environnements d'applications plus sécurisés et up-to-date. En outre, il peut s'avérer impossible pour vous de passer à des versions plus récentes de Kubernetes ou pour Amazon EKS d'installer des mises à jour de sécurité sur un cluster dégradé en raison de problèmes liés à l'infrastructure nécessaire ou à la configuration du cluster. Amazon EKS peut mettre 3 heures pour détecter les problèmes ou détecter qu'un problème est résolu.

La santé d'un cluster Amazon EKS est une responsabilité partagée entre Amazon EKS et ses utilisateurs. Vous êtes responsable de l'infrastructure préalable des rôles IAM et des sous-réseaux Amazon VPC, ainsi que des autres infrastructures nécessaires, qui doivent être fournies à l'avance. Amazon EKS détecte les modifications apportées à la configuration de cette infrastructure et du cluster.

Pour accéder à l'état de santé de votre cluster dans la console Amazon EKS, consultez la section intitulée Problèmes d'état dans l'onglet Présentation de la page détaillée du cluster Amazon EKS. Ces données sont également disponibles en appelant l'action `DescribeCluster` dans l'API EKS, par exemple depuis l' AWS Command Line Interface.

Pourquoi utiliser cette fonctionnalité ?

Vous bénéficierez d'une visibilité accrue sur l'état de santé de votre cluster Amazon EKS, diagnostiquerez et résoudrez rapidement les problèmes, sans avoir à passer du temps à déboguer ou à ouvrir des dossiers de AWS support. Par exemple, si vous avez accidentellement supprimé un sous-réseau pour le cluster Amazon EKS, Amazon EKS ne sera pas en mesure de créer des interfaces réseau et des Kubernetes AWS CLI commandes inter-comptes telles que `kubectl` exec ou `kubectl` logs. L'erreur suivante s'affiche : `Error from server: error dialing backend: remote error: tls: internal error.` Le problème d'état Amazon EKS indique : `subnet-da60e280 was deleted: could not create network interface`.

Comment cette fonctionnalité est-elle liée ou fonctionne-t-elle avec d'autres AWS services ?

Les rôles IAM et les sous-réseaux Amazon VPC sont deux exemples d'infrastructure préalable avec laquelle l'état du cluster détecte les problèmes. Cette fonctionnalité renvoie des informations détaillées si ces ressources ne sont pas correctement configurées.

Un cluster présentant des problèmes d'état entraîne-t-il des frais ?

Oui, chaque cluster Amazon EKS est facturé au tarif standard d'Amazon EKS. La fonctionnalité liée à l'état du cluster est disponible sans frais supplémentaires.

Cette fonctionnalité fonctionne-t-elle avec les clusters Amazon EKS sur AWS Outposts ?

Oui, des problèmes de cluster sont détectés pour les clusters EKS dans le AWS cloud, y compris les clusters étendus activés AWS Outposts et les clusters locaux activés AWS Outposts. L'état du cluster ne détecte pas les problèmes liés à Amazon EKS Anywhere ou à Amazon EKS Distro (EKS-D).

Puis-je être averti lorsque de nouveaux problèmes sont détectés ?

Non, vous devez vérifier la console Amazon EKS ou appeler l'API `DescribeCluster` EKS.

La console m'avertit-elle en cas de problème d'état ?

Oui, tout cluster présentant des problèmes d'état présentera une bannière en haut de la console.

Les deux premières colonnes sont celles qui sont nécessaires pour les valeurs de réponse de l'API. Le troisième champ de l' ClusterIssueobjet Health estresourceIds, dont le retour dépend du type de problème.

Code	Message	ResourceIds	Le cluster est-il récupérable ?	
SUBNET_NOT_FOUND	Nous n'avons pas pu trouver un ou plusieurs sous-réseaux actuellement associés à votre cluster. Appelez l'API update-cluster-config Amazon EKS pour mettre à jour les sous-réseaux.	ID de sous-réseaux	Oui	
SECURITY_GROUP_NOT_FOUND	Nous n'avons pas pu trouver un ou plusieurs groupes de sécurité actuellement associés à votre cluster. Appelez l' update-cluster-config API Amazon EKS pour mettre à jour les groupes de sécurité	ID de groupe de sécurité	Oui	
IP_NOT_AVAILABLE	Un ou plusieurs sous-réseaux associés à votre cluster ne disposent pas d'un nombre suffisant d'adresses IP pour qu'Amazon EKS puisse effectuer des opérations de gestion de cluster. Libérez des adresses dans le ou les sous-réseaux ou associez différents sous-réseaux à votre cluster à l'aide de l'API Amazon EKS update-cluster-config.	ID de sous-réseaux	Oui	
VPC_NOT_FOUND	Nous n'avons pas pu trouver le VPC associé à votre cluster. Vous devez supprimer et recréer votre cluster.	ID du VPC	Non	

Code	Message	ResourceIds	Le cluster est-il récupérable ?	
ASSUME_ROLE_ACCESS_DENIED	Votre cluster n'utilise pas Amazon EKS service-linked-role. Nous ne pouvions pas assumer le rôle associé à votre cluster pour effectuer les opérations de gestion Amazon EKS requises. Vérifiez que le rôle existe et qu'il dispose de la politique de confiance requise.	Rôle IAM du cluster	Oui	
PERMISSION_ACCESS_DENIED	Votre cluster n'utilise pas Amazon EKS service-linked-role. Le rôle associé à votre cluster n'accorde pas les autorisations suffisantes à Amazon EKS pour effectuer les opérations de gestion requises. Vérifiez les politiques associées au rôle de cluster et si des politiques de refus distinctes sont appliquées.	Rôle IAM du cluster	Oui	
ASSUME_ROLE_ACCESS_DENIED_USING_SLR	Nous ne pouvions pas assumer la gestion du cluster Amazon EKS service-linked-role. Vérifiez que le rôle existe et qu'il dispose de la politique de confiance requise.	L'Amazon EKS service-linked-role	Oui	

Code	Message	ResourceIds	Le cluster est-il récupérable ?
PERMISSION_ACCESS_DENIED_USING_SLR	La gestion du cluster Amazon EKS service-linked-role n'accorde pas d'autorisations suffisantes à Amazon EKS pour effectuer les opérations de gestion requises. Vérifiez les politiques associées au rôle de cluster et si des politiques de refus distinctes sont appliquées.	L'Amazon EKS service-linked-role	Oui
OPT_IN_REQUIRED	Votre compte ne possède pas d'abonnement au service Amazon EC2. Mettez à jour les abonnements de votre compte sur la page des paramètres de votre compte.	N/A	Oui
STS_REGIONAL_ENDPOINT_DISABLED	Le point de terminaison STS régional est désactivé. Activez le point de terminaison pour qu'Amazon EKS effectue les opérations de gestion de cluster requises.	N/A	Oui
KMS_KEY_DISABLED	La AWS KMS clé associée à votre cluster est désactivée. Réactivez la clé pour récupérer votre cluster.	L'interface KMS Key Arn	Oui
KMS_KEY_NOT_FOUND	Nous n'avons pas trouvé la AWS KMS clé associée à votre cluster. Vous devez supprimer et recréer le cluster.	L'interface KMS Key ARN	Non

Code	Message	ResourceIds	Le cluster est-il récupérable ?	
KMS_GRANT_REVOKED	Les autorisations pour la AWS KMS clé associée à votre cluster sont révoquées. Vous devez supprimer et recréer le cluster.	L'interface KMS Key Arn	Non	

Amazon EKS Connector

Vous pouvez utiliser Amazon EKS Connector pour enregistrer et connecter tout cluster Kubernetes conforme à AWS et le visualiser dans la console Amazon EKS. Une fois qu'un cluster est connecté, vous pouvez voir le statut, la configuration et les charges de travail de ce cluster dans la console Amazon EKS. Vous pouvez utiliser cette fonction pour afficher les clusters connectés dans la console Amazon EKS, mais vous ne pouvez pas les gérer. Le connecteur Amazon EKS est également un agent qui est un projet open source sur Github. Pour obtenir du contenu technique supplémentaire, y compris les questions fréquentes et le dépannage, veuillez consulter Résolution des problèmes dans Amazon EKS Connector

Le connecteur Amazon EKS peut connecter les types de clusters Kubernetes suivants à Amazon EKS.

- Clusters Kubernetes sur site
- Clusters autogérés qui s'exécutent sur Amazon EC2
- Clusters gérés par d'autres fournisseurs de cloud

Considérations relatives à Amazon EKS Connector

Avant d'utiliser Amazon EKS Connector, comprenez les points suivants :

- Vous devez disposer des privilèges administratifs sur le cluster Kubernetes pour connecter le cluster à Amazon EKS.
- Le cluster Kubernetes doit disposer de composants master Linux 64 bits (x86) avant de se connecter. Les composants master ARM ne sont pas pris en charge.
- Vous devez disposer de composants master dans votre cluster Kubernetes qui ont un accès sortant aux points de terminaison Systems Manager `ssm.` et `ssmmessages.`. Pour de plus amples informations, veuillez consulter points de terminaison Systems Manager dans Références générales AWS.
- Par défaut, vous pouvez connecter jusqu'à 10 clusters dans une région. Vous pouvez demander une augmentation par le biais de la console de quota de service. Consultez Demander une augmentation de quota pour plus d'informations.
- Seules les API `RegisterCluster`, `ListClusters`, `DescribeCluster` et `DeregisterCluster` Amazon EKS sont prises en charge pour les clusters Kubernetes externes.

- Vous devez disposer des autorisations suivantes pour enregistrer un cluster :
  - eks:RegisterCluster
  - ssm:CreateActivation
  - ssm:DeleteActivation
  - iam:PassRole
- Vous devez disposer des autorisations suivantes pour annuler l'enregistrement d'un cluster :
  - eks:DeregisterCluster
  - ssm:DeleteActivation
  - ssm:DeregisterManagedInstance

Rôles IAM requis pour Amazon EKS Connector

L'utilisation du Amazon EKS Connector nécessite les deux rôles IAM suivants :

- Le rôle lié à un service du Amazon EKS Connector est créé lorsque vous enregistrez un cluster pour la première fois.
- Vous devez créer le rôle IAM de l'agent Amazon EKS Connector. Consultez Rôle IAM du connecteur Amazon EKS pour plus de détails.

Pour activer l'autorisation d'affichage du cluster et de la charge de travail pour des principaux IAM, appliquez le `eks-connector` et les rôles de cluster Amazon EKS Connector à votre cluster. Suivez les étapes de Octroi d'un accès à un principal IAM pour afficher les ressources Kubernetes sur un cluster.

Connexion d'un cluster externe

Vous pouvez connecter un cluster Kubernetes externe vers Amazon EKS en utilisant plusieurs méthodes dans le cadre du processus suivant. Ce processus comporte deux étapes : enregistrement du cluster auprès d'Amazon EKS et installation de l'agent `eks-connector` dans le cluster.

> **Important**
>
> Vous devez effectuer la deuxième étape dans les 3 jours suivant la fin de la première étape, avant l'expiration de l'enregistrement.

Méthodes de connexion

Toutes les méthodes d'installation de l'agent ne peuvent pas être utilisées après chacune des méthodes d'enregistrement du cluster. Le tableau suivant répertorie chacune des méthodes d'enregistrement et les méthodes d'installation de l'agent qui peuvent être utilisées.

Étape	Méthodes		
Enregistrer le cluster	AWS Management Console	AWS Command Line Interface	`eksctl`
Installer l'agent	Helm, manifestes YAML	Helm, manifestes YAML	Manifestes YAML

Prérequis

- Assurez-vous que le rôle d'agent Amazon EKS Connector est créé. Suivez les étapes de Création du rôle de l'agent Amazon EKS Connector.
- Vous devez disposer des autorisations suivantes pour enregistrer un cluster :
  - `eks:RegisterCluster`
  - `ssm:CreateActivation`
  - `ssm:DeleteActivation`
  - `iam:PassRole`

Étape 1 : Enregistrement du cluster

AWS CLI

Prérequis

- La AWS CLI doit être installée. Pour l'installer ou la mettre à niveau, consultez Installation de la AWS CLI.

Pour inscrire votre cluster dans le serveur AWS CLI

- Pour la configuration du connecteur, spécifiez votre rôle IAM d'agent Amazon EKS Connector. Pour plus d'informations, consultez Rôles IAM requis pour Amazon EKS Connector.

```
aws eks register-cluster \
     --name my-first-registered-cluster \
     --connector-config roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider="OTHER" \
     --region aws-region
```

L'exemple qui suit illustre un résultat.

```
{
    "cluster": {
        "name": "my-first-registered-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-first-registered-
cluster",
        "createdAt": 1627669203.531,
        "ConnectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": 1627672543.0,
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole"
        },
        "status": "CREATING"
    }
}
```

Vous utilisez les valeurs `aws-region`, `activationId` et `activationCode` à l'étape suivante.

AWS Management Console

Pour enregistrer votre cluster Kubernetes dans la console.

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.

2. Sélectionnez Add cluster (Ajouter un cluster), puis Register (Enregistrer) pour afficher la page de configuration.
3. Sur la page Configurer le cluster, renseignez les champs suivants :
   - Nom : nom unique de votre cluster.
   - Fournisseur : sélectionnez cet élément pour afficher la liste déroulante des fournisseurs de cluster Kubernetes. Si vous ne connaissez pas le fournisseur spécifique, sélectionnez Autre.
   - Rôle EKS Connector : sélectionnez le rôle à utiliser pour connecter le cluster.
4. Sélectionnez Register cluster (Enregistrer le cluster).
5. La page Présentation du cluster s'affiche. Si vous souhaitez utiliser les Charts de Helm, copiez la commande `helm install` et passez à l'étape suivante. Si vous souhaitez utiliser le manifeste YAML, sélectionnez Télécharger le fichier YAML pour télécharger le fichier manifeste sur votre lecteur local.

   > **Important**
   >
   > - C'est le seul moment où vous pouvez copier la commande `helm install` ou télécharger ce fichier. Ne naviguez pas hors de cette page, car le lien ne sera pas accessible et vous devrez annuler l'enregistrement du cluster et recommencer les étapes depuis le début.
   > - La commande ou le fichier manifeste ne peut être utilisé(e) qu'une seule fois pour le cluster enregistré. Si vous supprimez des ressources du cluster Kubernetes, vous devez réenregistrer le cluster et obtenir un nouveau fichier manifeste.

   Passez à l'étape suivante pour appliquer le fichier manifeste à votre cluster Kubernetes.

eksctl

Prérequis

- `eksctl` version `0.68` ou une version ultérieure doit être installée. Pour l'installer ou la mettre à niveau, veuillez consulter la rubrique Démarrage avec Amazon EKS : `eksctl`.

Pour inscrire votre cluster avec **eksctl**

1. Enregistrez le cluster en fournissant un nom, un fournisseur et une région.

```
eksctl register cluster --name my-cluster --provider my-provider --
region region-code
```

Exemple de sortie :

```
2021-08-19 13:47:26 [#]  creating IAM role "eksctl-20210819194112186040"
2021-08-19 13:47:26 [#]  registered cluster "<name>" successfully
2021-08-19 13:47:26 [#]  wrote file eks-connector.yaml to <current directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-clusterrole.yaml to <current
 directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-console-dashboard-full-access-
group.yaml to <current directory>
2021-08-19 13:47:26 [!]  note: "eks-connector-clusterrole.yaml" and "eks-
connector-console-dashboard-full-access-group.yaml" give full EKS Console access
 to IAM identity "<aws-arn>", edit if required; read https://eksctl.io/usage/
eks-connector for more info
2021-08-19 13:47:26 [#]  run `kubectl apply -f eks-connector.yaml,eks-connector-
clusterrole.yaml,eks-connector-console-dashboard-full-access-group.yaml` before
 expiry> to connect the cluster
```

Cela crée des fichiers sur votre ordinateur local. Ces fichiers doivent être appliqués au cluster externe dans un délai de 3 jours, sinon l'enregistrement expire.

2. Dans un terminal pouvant accéder au cluster, appliquez le fichier `eks-connector-binding.yaml` :

```
kubectl apply -f eks-connector-binding.yaml
```

Étape 2 : Installation de l'agent **eks-connector**

Helm chart

1. Si vous avez utilisé la AWS CLI à l'étape précédente, remplacez le `ACTIVATION_CODE` et l'`ACTIVATION_ID` dans la commande suivante avec les

valeurs `activationId` et `activationCode` respectivement. Remplacez la `aws-region` par la Région AWS que vous avez utilisée à l'étape précédente. Ensuite, exécutez la commande pour installer l'agent `eks-connector` dans le cluster d'enregistrement :

```
$ helm install eks-connector \
  --namespace eks-connector \
  oci://public.ecr.aws/eks-connector/eks-connector-chart \
  --set eks.activationCode=ACTIVATION_CODE \
  --set eks.activationId=ACTIVATION_ID \
  --set eks.agentRegion=aws-region
```

Si vous avez utilisé la AWS Management Console à l'étape précédente, utilisez la commande que vous avez copiée à l'étape précédente et qui contient ces valeurs.

2. Vérifiez l'état de santé du déploiement du `eks-connector` installé et attendez que le statut du cluster enregistré dans Amazon EKS soit `ACTIVE`.

YAML manifest

Effectuez la connexion en appliquant le fichier manifeste Amazon EKS Connector à votre cluster Kubernetes. Pour ce faire, vous devez utiliser les méthodes décrites précédemment. Si le manifeste n'est pas appliqué dans un délai de trois jours, l'enregistrement du connecteur Amazon EKS expire. Si la connexion de cluster expire, le cluster doit être désenregistré avant de reconnecter le cluster.

1. Téléchargez le fichier YAML Amazon EKS Connector.

```
curl -O https://amazon-eks.s3.us-west-2.amazonaws.com/eks-connector/manifests/
eks-connector/latest/eks-connector.yaml
```

2. Modifiez le fichier YAML Amazon EKS Connector pour remplacer toutes les références de `%AWS_REGION%`, `%EKS_ACTIVATION_ID%`, `%EKS_ACTIVATION_CODE%` par la `aws-region`, l'`activationId` et le `activationCode` à partir de la sortie de l'étape antérieure.

   L'exemple de commande suivant peut remplacer ces valeurs.

```
sed -i "s~%AWS_REGION%~$aws-region~g; s~%EKS_ACTIVATION_ID
%~$EKS_ACTIVATION_ID~g; s~%EKS_ACTIVATION_CODE%~$(echo -n $EKS_ACTIVATION_CODE |
 base64)~g" eks-connector.yaml
```

> **Important**
>
> Assurez-vous que votre code d'activation est au format base64.

3. Dans un terminal pouvant accéder au cluster, vous pouvez appliquer le fichier manifeste mis à jour en exécutant la commande suivante :

```
kubectl apply -f eks-connector.yaml
```

4. Après l'application du manifeste Amazon EKS Connector et des fichiers YAML de liaison de rôle à votre cluster Kubernetes, vérifiez que le cluster est maintenant connecté.

```
aws eks describe-cluster \
     --name "my-first-registered-cluster" \
     --region AWS_REGION
```

   La sortie doit inclure `status=ACTIVE`.

5. (Facultatif) Attribuez des balises à votre cluster. Pour plus d'informations, consultez Étiquetage de vos ressources Amazon EKS.

Étapes suivantes

Si vous rencontrez des problèmes lors de ces étapes, consultez la rubrique Résolution des problèmes dans Amazon EKS Connector.

Pour permettre à des principaux IAM d'accéder à la console Amazon EKS pour afficher ressources Kubernetes d'un cluster connecté, consultez la rubrique Octroi d'un accès à un principal IAM pour afficher les ressources Kubernetes sur un cluster.

Octroi d'un accès à un principal IAM pour afficher les ressources Kubernetes sur un cluster

Accordez à des principaux IAM l'accès à la console Amazon EKS pour afficher des informations sur les ressources Kubernetes exécutées sur votre cluster connecté.

Prérequis

Le principal IAM que vous utilisez pour accéder à la AWS Management Console doit répondre aux exigences suivantes :

- Il doit bénéficier de l'autorisation IAM d'`eks:AccessKubernetesApi`.
- Le compte de service Amazon EKS Connector peut emprunter l'identité du principal IAM dans le cluster. Cela permet au service Amazon EKS Connector de mapper le principal IAM à un utilisateur Kubernetes.

Pour créer et appliquer le rôle de cluster Amazon EKS Connector

1. Téléchargez le modèle de rôle de cluster `eks-connector`.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-clusterrole.yaml
   ```

2. Modifiez le fichier YAML du modèle de rôle de cluster. Remplacez les références de `%IAM_ARN%` par l'Amazon Resource Name (ARN) de votre principal IAM.
3. Appliquez le fichier YAML du rôle du cluster Amazon EKS Connector à votre cluster Kubernetes.

   ```
   kubectl apply -f eks-connector-clusterrole.yaml
   ```

Pour qu'un principal IAM puisse afficher les ressources Kubernetes dans la console Amazon EKS, il doit être associé à un `clusterrole` ou un Kubernetes `role` disposant des autorisations nécessaires pour lire ces ressources. Pour plus d'informations, consultez Utilisation de l'autorisation RBAC dans la documentation Kubernetes.

Pour configurer un principal IAM pour accéder au cluster connecté

1. Vous pouvez télécharger n'importe lequel de ces exemples de fichier manifeste pour créer un `clusterrole` et une `clusterrolebinding` ou un `role` et une `rolebinding` :

   Afficher les ressources Kubernetes dans tous les espaces de noms

   Le rôle de cluster `eks-connector-console-dashboard-full-access-clusterrole` donne accès à tous les espaces de noms et toutes les ressources qui peuvent être visualisés dans la console. Vous pouvez changer le nom du rôle `role`, du

`clusterrole` et de leur liaison correspondante avant de l'appliquer à votre cluster. Utilisez la commande suivante pour télécharger un exemple de fichier.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/
eks-connector-console-roles/eks-connector-console-dashboard-full-access-
group.yaml
```

Afficher les ressources Kubernetes dans un espace de noms spécifique

L'espace de noms dans ce fichier est `default`. Par conséquent, si vous voulez spécifier un espace de noms différent, éditez le fichier avant de l'appliquer à votre cluster. Utilisez la commande suivante pour télécharger un exemple de fichier.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/
eks-connector-console-roles/eks-connector-console-dashboard-restricted-access-
group.yaml
```

2. Modifiez le fichier YAML à accès complet ou à accès restreint pour remplacer les références de `%IAM_ARN%` par l'Amazon Resource Name (ARN) de votre principal IAM.
3. Appliquez les fichiers YAML à accès complet ou à accès restreint à votre cluster Kubernetes. Remplacez la valeur du fichier YAML par la vôtre.

```
kubectl apply -f eks-connector-console-dashboard-full-access-group.yaml
```

Pour afficher les ressources Kubernetes dans votre cluster connecté, consultez Afficher les ressources Kubernetes. Les données relatives à certains types de ressources de l'onglet Ressources ne sont pas disponibles pour les clusters connectés.

Annulation de l'enregistrement d'un cluster

Si vous n'utilisez plus un cluster, vous pouvez annuler son enregistrement. Après l'annulation de son enregistrement, le cluster n'est plus visible dans la console Amazon EKS.

Vous devez disposer des autorisations suivantes pour appeler l'API deregisterCluster :

- `eks:DeregisterCluster`
- `ssm:DeleteActivation`
- `ssm:DeregisterManagedInstance`

Ce processus comporte deux étapes : annulation de l'enregistrement du cluster auprès d'Amazon EKS et désinstallation de l'agent eks-connector dans le cluster.

Pour annuler l'enregistrement du cluster Kubernetes

AWS CLI

Prérequis

- La AWS CLI doit être installée. Pour l'installer ou la mettre à niveau, consultez Installation de la AWS CLI.
- Assurez-vous que le rôle d'agent Amazon EKS Connector est créé.

Annulez l'enregistrement du cluster connecté.

```
aws eks deregister-cluster \
    --name my-cluster \
    --region region-code
```

AWS Management Console

1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
2. Choisissez Clusters.
3. Sur la page Clusters, sélectionnez le cluster connecté et sélectionnez Deregister (Annuler l'enregistrer).
4. Confirmez que vous voulez annuler l'enregistrement du cluster.

eksctl

Prérequis

- `eksctl` version `0.68` ou une version ultérieure doit être installée. Pour l'installer ou la mettre à niveau, veuillez consulter la rubrique Démarrage avec Amazon EKS : `eksctl`.
- Assurez-vous que le rôle d'agent Amazon EKS Connector est créé.

Pour annuler l'enregistrement de votre cluster avec **eksctl**

- Pour la configuration du connecteur, spécifiez votre rôle IAM d'agent Amazon EKS Connector. Pour plus d'informations, consultez Rôles IAM requis pour Amazon EKS Connector.

```
eksctl deregister cluster --name my-cluster
```

Pour nettoyer les ressources dans votre cluster Kubernetes

Helm

- Exécutez la commande suivante pour désinstaller l'agent.

```
helm -n eks-connector uninstall eks-connector
```

YAML manifest

1. Supprimez le fichier YAML Amazon EKS Connector de votre cluster Kubernetes.

```
kubectl delete -f eks-connector.yaml
```

2. Si vous avez créé un `clusterrole` ou des `clusterrolebindings` pour des principaux IAM supplémentaires pour accéder au cluster, veillez à les supprimer de votre cluster Kubernetes.

Résolution des problèmes dans Amazon EKS Connector

Cette rubrique couvre certaines des erreurs les plus courantes que vous pouvez rencontrer lors de l'utilisation d'Amazon EKS Connector, y compris des instructions sur la façon de les résoudre et des solutions de contournement.

Dépannage de base

Cette section décrit les étapes à suivre pour diagnostiquer le problème s'il n'est pas clair.

Vérifier le statut d'Amazon EKS Connector

Vérifiez le statut d'Amazon EKS Connector.

```
kubectl get pods -n eks-connector
```

Inspecter les journaux d'Amazon EKS Connector

Le Pod Amazon EKS Connector se compose de trois conteneurs. Pour récupérer les journaux complets de tous ces conteneurs afin que vous puissiez les inspecter, exécutez les commandes suivantes :

- `connector-init`

  ```
  kubectl logs eks-connector-0 --container connector-init -n eks-connector
  kubectl logs eks-connector-1 --container connector-init -n eks-connector
  ```

- `connector-proxy`

  ```
  kubectl logs eks-connector-0 --container connector-proxy -n eks-connector
  kubectl logs eks-connector-1 --container connector-proxy -n eks-connector
  ```

- `connector-agent`

  ```
  kubectl exec eks-connector-0 --container connector-agent -n eks-connector -- cat /
  var/log/amazon/ssm/amazon-ssm-agent.log
  kubectl exec eks-connector-1 --container connector-agent -n eks-connector -- cat /
  var/log/amazon/ssm/amazon-ssm-agent.log
  ```

Obtenir le nom effectif du cluster

Les clusters Amazon EKS sont identifiés de manière unique par `clusterName` dans un seul compte AWS et une seule Région AWS. Si vous avez plusieurs clusters connectés dans Amazon EKS, vous pouvez confirmer auprès de quel cluster Amazon EKS le cluster Kubernetes actuel est enregistré. Pour ce faire, saisissez la commande suivante pour connaître le `clusterName` du cluster actuel.

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
kubectl exec eks-connector-1 --container connector-agent -n eks-connector \
```

```
 -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
| sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
```

Commandes diverses

Les commandes suivantes sont utiles pour récupérer les informations dont vous avez besoin pour résoudre les problèmes.

- Utilisez la commande suivante pour collecter des images utilisées par les Pods dans Amazon EKS Connector.

  ```
  kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.containers[*].image}"
   | tr -s '[[:space:]]' '\n'
  ```

- Utilisez la commande suivante pour rassembler les noms des nœuds sur lesquels Amazon EKS Connector est exécuté.

  ```
  kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.nodeName}" | tr -s
   '[[:space:]]' '\n'
  ```

- Exécutez la commande suivante pour obtenir les versions de votre client et serveur Kubernetes.

  ```
  kubectl version
  ```

- Exécutez la commande suivante pour obtenir des informations sur vos nœuds.

  ```
  kubectl get nodes -o wide --show-labels
  ```

Problème Helm : 403 Forbidden

Si vous avez reçu le message d'erreur suivant lors de l'exécution des commandes d'installation de Helm :

```
Error: INSTALLATION FAILED: unexpected status from HEAD request to https://
public.ecr.aws/v2/eks-connector/eks-connector-chart/manifests/0.0.6: 403 Forbidden
```

Vous pouvez exécuter la ligne suivante pour y remédier :

```
docker logout public.ecr.aws
```

Erreur de console : le cluster est bloqué dans l'état En attente

Si le cluster reste bloqué dans son `Pending` état sur la console Amazon EKS une fois que vous l'avez enregistré, cela peut être dû au fait que le connecteur Amazon EKS n'a pas AWS encore réussi à connecter le cluster au cluster. Pour un cluster enregistré, l'état `Pending` indique que la connexion n'a pas encore été établie avec succès. Pour résoudre ce problème, assurez-vous d'avoir appliqué le manifeste au cluster Kubernetes cible. Si vous l'avez appliqué au cluster mais que celui-ci est toujours à l'état `Pending`, il est fort probable que le statefulset `eks-connector` ne soit pas sain. Pour résoudre ce problème, consultez Les Pods du connecteur Amazon EKS plantent et redémarrent, indéfiniment dans cette rubrique.

Erreur de console : **`User "system:serviceaccount:eks-connector:eks-connector" can't impersonate resource "users" in API group ""`** au niveau du cluster

Amazon EKS Connector utilise l'emprunt d'identité de l'utilisateur de Kubernetes pour effectuer des opérations au nom des principaux IAM depuis la AWS Management Console. Chaque principal qui accède à l'KubernetesAPI depuis le compte de AWS `eks-connector` service doit être autorisé à se faire passer pour l'Kubernetesutilisateur correspondant avec un ARN IAM comme nom d'utilisateur. Kubernetes Dans les exemples suivants, l'ARN IAM est mappé à un utilisateur Kubernetes.

- L'utilisateur IAM *`john`* du AWS compte *`111122223333`* est mappé à un Kubernetes utilisateur. Les bonnes pratiques IAM recommandent d'accorder des autorisations à des rôles plutôt qu'à des utilisateurs.

  ```
  arn:aws:iam::111122223333:user/john
  ```

- Le rôle IAM *`admin`* du AWS compte *`111122223333`* est mappé à un Kubernetes utilisateur :

  ```
  arn:aws:iam::111122223333:role/admin
  ```

  Le résultat est un ARN de rôle IAM, au lieu de l'ARN de AWS STS session.

Pour savoir comment configurer `ClusterRole` et `ClusterRoleBinding` afin d'accorder au compte de service `eks-connector` le privilège de compte de service pour usurper l'identité de l'utilisateur mappé, consultez Octroi d'un accès à un principal IAM pour afficher les ressources Kubernetes sur

un cluster. Assurez-vous que, dans le modèle, `%IAM_ARN%` est remplacé par l'ARN IAM du principal IAM de la AWS Management Console .

Erreur de console : **`[...] is forbidden: User [...] cannot list resource "[...] in API group"`** au niveau du cluster

Considérons le problème suivant. Le connecteur Amazon EKS s'est fait passer pour le principal AWS Management Console IAM demandeur dans le cluster cible. Kubernetes Toutefois, le principal ne dispose d'aucune autorisation RBAC pour les opérations d'API Kubernetes.

Pour résoudre ce problème, il existe deux méthodes pour accorder des autorisations à des utilisateurs supplémentaires. Si vous avez déjà installé eks-connector via les charts de Helm, vous pouvez facilement accorder l'accès aux utilisateurs en exécutant la commande suivante. Remplacez les `userARN1` et `userARN2` avec une liste des ARN des rôles IAM pour donner accès à la consultation des ressources Kubernetes :

```
helm upgrade eks-connector oci://public.ecr.aws/eks-connector/eks-connector-chart \
    --reuse-values \
    --set 'authentication.allowedUserARNs={userARN1,userARN2}'
```

Ou, en tant qu'administrateur du cluster, accordez le niveau approprié de privilèges RBAC aux différents utilisateurs Kubernetes. Pour plus d'informations et d'exemples, consultez Octroi d'un accès à un principal IAM pour afficher les ressources Kubernetes sur un cluster.

Erreur de console : Amazon EKS ne peut pas communiquer avec le serveur d'API de votre cluster Kubernetes. Le cluster doit être dans un état ACTIF pour que la connexion soit réussie. Veuillez réessayer dans quelques minutes.

Si le service Amazon EKS ne peut pas communiquer avec Amazon EKS Connector dans le cluster cible, cela peut être dû à l'une des raisons suivantes :

- Amazon EKS Connector dans le cluster cible n'est pas sain.
- Une connectivité médiocre ou une connexion interrompue entre le cluster cible et la Région AWS.

Pour résoudre ce problème, consultez les journaux d'Amazon EKS Connector. Si vous ne voyez pas d'erreur sur Amazon EKS Connector, réessayez la connexion après quelques minutes. Si vous êtes

régulièrement confronté à une latence élevée ou à une connectivité intermittente pour le cluster cible, envisagez de le réenregistrer dans un Région AWS cluster situé plus près de chez vous.

Les Pods du connecteur Amazon EKS plantent et redémarrent, indéfiniment

De nombreuses raisons peuvent amener un Pod Amazon EKS Connector à passer à l'état `CrashLoopBackOff`. Ce problème concerne probablement le conteneur `connector-init`. Vérifiez l'état du Pod Amazon EKS Connector.

```
kubectl get pods -n eks-connector
```

L'exemple qui suit illustre un résultat.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:CrashLoopBackOff   1          7s
```

Si votre sortie est similaire à la sortie précédente, consultez la section Inspecter les journaux d'Amazon EKS Connector pour résoudre le problème.

`Failed to initiate eks-connector: InvalidActivation`

Lorsque vous démarrez Amazon EKS Connector pour la première fois, il enregistre un `activationId` et `activationCode` avec Amazon Web Services. L'enregistrement peut échouer, ce qui peut provoquer le crash du conteneur `connector-init` avec une erreur similaire à la suivante.

```
F1116 20:30:47.261469       1 init.go:43] failed to initiate eks-connector:
 InvalidActivation:
```

Pour résoudre ce problème, tenez compte des causes suivantes et des correctifs recommandés :

- L'enregistrement a peut-être échoué car `activationId` et `activationCode` ne se trouvent pas dans votre fichier manifeste. Si c'est le cas, assurez-vous qu'il s'agit des valeurs correctes renvoyées par l'opération d'API `RegisterCluster` et que `activationCode` se trouve dans le fichier manifeste. `activationCode` est ajouté aux secrets Kubernetes, il doit donc être codé au format `base64`. Pour plus d'informations, consultez Étape 1 : Enregistrement du cluster.
- L'enregistrement a peut-être échoué car votre activation a expiré. En effet, pour des raisons de sécurité, vous devez activer Amazon EKS Connector dans les trois jours suivant l'enregistrement

du cluster. Pour résoudre ce problème, assurez-vous que le manifeste Amazon EKS Connector est appliqué au cluster Kubernetes cible avant la date et l'heure d'expiration. Pour confirmer la date d'expiration de votre activation, exécutez un appel à l'opération API `DescribeCluster`.

```
aws eks describe-cluster --name my-cluster
```

Dans l'exemple de réponse suivant, la date et l'heure d'expiration sont enregistrées sous la forme `2021-11-12T22:28:51.101000-08:00`.

```
{
    "cluster": {
        "name": "my-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-cluster",
        "createdAt": "2021-11-09T22:28:51.449000-08:00",
        "status": "FAILED",
        "tags": {
        },
        "connectorConfig": {
            "activationId": "00000000-0000-0000-0000-000000000000",
            "activationExpiry": "2021-11-12T22:28:51.101000-08:00",
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/my-connector-role"
        }
    }
}
```

Si l'`activationExpiry` est passée, désenregistrez le cluster et enregistrez-le à nouveau. Cela génère une nouvelle activation.

Le nœud du cluster manque de connectivité sortante

Pour fonctionner correctement, Amazon EKS Connector nécessite une connectivité sortante à plusieurs points de terminaison AWS . Vous ne pouvez pas connecter un cluster privé sans connectivité sortante à une Région AWS cible. Pour résoudre ce problème, vous devez ajouter la connectivité sortante nécessaire. Pour plus d'informations sur les exigences de connecteur, consultez Considérations relatives à Amazon EKS Connector.

Les Pods Amazon EKS Connector sont dans l'état `ImagePullBackOff`

Si vous exécutez la commande `get pods` et que les Pods sont dans l'état `ImagePullBackOff`, ils ne peuvent pas fonctionner correctement. Si les Pods Amazon EKS Connector sont dans l'état `ImagePullBackOff`, ils ne peuvent pas fonctionner correctement. Vérifiez l'état de vos Pods Amazon EKS Connector.

```
kubectl get pods -n eks-connector
```

L'exemple qui suit illustre un résultat.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:ImagePullBackOff   0          4s
```

Le fichier manifeste Amazon EKS Connector par défaut fait référence aux images de la galerie publique Amazon ECR. Il est possible que le cluster Kubernetes cible ne puisse pas extraire les images de la galerie publique Amazon ECR. Veuillez soit résoudre le problème d'extraction d'images de la galerie publique Amazon ECR, soit envisager la mise en miroir des images dans le registre de conteneurs privé de votre choix.

Questions fréquentes (FAQ)

Q : Comment fonctionne la technologie sous-jacente à Amazon EKS Connector ?

R : Amazon EKS Connector est basé sur l'agent AWS Systems Manager (System Manager). Amazon EKS Connector s'exécute en tant que `StatefulSet` sur votre cluster Kubernetes. Il établit une connexion et proxie la communication entre le serveur API de votre cluster et Amazon Web Services. Cela permet d'afficher les données du cluster dans la console Amazon EKS jusqu'à ce que vous déconnectiez le cluster de AWS. L'agent Systems Manager est un projet open source. Pour plus d'informations sur ce projet, consultez la page du projet sur GitHub.

Q : j'ai un cluster Kubernetes sur site que je souhaite connecter. Dois-je ouvrir des ports de pare-feu pour le connecter ?

R : Non, vous n'avez pas besoin d'ouvrir de ports de pare-feu. Le cluster Kubernetes n'a besoin que d'une connexion sortante vers les Régions AWS. Les services AWS n'accèdent jamais aux ressources de votre réseau sur site. Amazon EKS Connector s'exécute sur votre cluster et initie la connexion à AWS. Lorsque l'enregistrement du cluster est terminé, AWS n'émet des commandes

vers Amazon EKS Connector qu'après avoir lancé une action depuis la console Amazon EKS qui nécessite des informations du serveur API Kubernetes sur votre cluster.

Q : Quelles sont les données envoyées de mon cluster à AWS par Amazon EKS Connector ?

R : Amazon EKS Connector envoie les informations techniques nécessaires à l'enregistrement de votre cluster sur AWS. Il envoie également des métadonnées de cluster et de charge de travail pour les fonctions de la console Amazon EKS demandées par les clients. Amazon EKS Connector ne recueille ou n'envoie ces données que si vous lancez une action à partir de la console Amazon EKS ou de l'API Amazon EKS qui nécessite l'envoi de ces données à AWS. À part le numéro de version de Kubernetes, AWS ne stocke aucune donnée par défaut. Il ne stocke les données que si vous l'y autorisez.

Q : Puis-je connecter un cluster en dehors d'une Région AWS ?

R : Oui, vous pouvez connecter un cluster depuis n'importe quel emplacement à Amazon EKS. De plus, votre service Amazon EKS peut être situé dans n'importe quelle Région AWScommerciale publique AWS. Cela fonctionne avec une connexion réseau valide entre votre cluster et la Région AWS cible. Nous vous recommandons de choisir une Région AWS la plus proche de l'emplacement de votre cluster pour optimiser les performances de l'interface utilisateur. Par exemple, si vous avez un cluster exécuté à Tokyo, connectez votre cluster à la Région AWS de Tokyo (c'est-à-dire la Région AWS `ap-northeast-1`) pour réduire la latence. Vous pouvez connecter un cluster depuis n'importe quel emplacement à Amazon EKS dans n'importe quelle Régions AWS commerciale publique, à l'exception des Régions AWS Chine ou GovCloud.

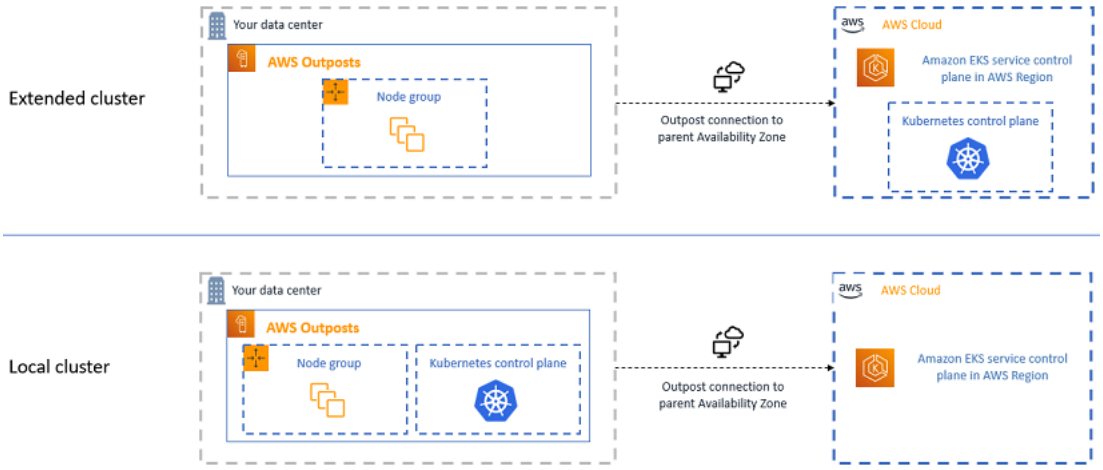
Amazon EKS sur AWS Outposts

Vous pouvez utiliser Amazon EKS pour exécuter des applications Kubernetes sur site sur AWS Outposts. Vous pouvez déployer Amazon EKS sur des Outposts avec :

- Des clusters étendus – Exécutez le plan de contrôle Kubernetes dans une Région AWS et les nœuds sur votre Outpost.
- Des clusters locaux – Exécutez le plan de contrôle Kubernetes et les nœuds sur votre Outpost.

Pour les deux options de déploiement, le plan de contrôle Kubernetes est entièrement géré par AWS. Vous pouvez utiliser les API, les outils et la console Amazon EKS que vous utilisez dans le cloud pour créer et exécuter Amazon EKS sur des Outposts.

Le schéma suivant illustre ces options de déploiement.



Quand utiliser chaque option de déploiement

Les clusters locaux et étendus sont des options de déploiement polyvalentes et peuvent être utilisées pour de nombreuses applications.

Avec les clusters locaux, vous pouvez exécuter l'intégralité du cluster Amazon EKS localement sur des Outposts. Cette option permet de réduire le risque d'interruption des applications qui pourrait résulter de déconnexions temporaires du réseau au cloud. Ces déconnexions du réseau peuvent être causées par des coupures de fibre ou des événements météorologiques. Étant donné que l'ensemble du cluster Amazon EKS s'exécute localement sur les Outposts, les applications restent disponibles. Vous pouvez effectuer des opérations de cluster en cas de déconnexion du réseau au cloud. Pour

de plus amples informations, veuillez consulter Préparation aux déconnexions du réseau. Si vous êtes préoccupé par la qualité de la connexion réseau entre vos Outposts et la Région AWS parent et que vous avez besoin d'une disponibilité élevée en cas de déconnexions du réseau, utilisez le cluster local comme option de déploiement.

Avec les clusters étendus, vous pouvez conserver la capacité de votre Outpost, car le plan de contrôle Kubernetes s'exécute dans la Région AWS parent. Cette option est adaptée si vous pouvez investir dans une connectivité réseau fiable et redondante entre votre Outpost et la Région AWS. La qualité de la connexion réseau est essentielle pour cette option. La façon dont Kubernetes gère les déconnexions du réseau entre le plan de contrôle Kubernetes et les nœuds peut entraîner une interruption de l'application. Pour plus d'informations sur le comportement de Kubernetes, consultez Scheduling, Preemption, and Eviction (langue française non garantie) dans la documentation Kubernetes.

Comparaison des options de déploiement

Le tableau suivant compare les différences entre les deux options.

Fonctionnalité	Cluster étendu	Cluster local
Emplacement du plan de contrôle Kubernetes	Région AWS	Outpost
Compte du plan de contrôle Kubernetes	Compte AWS	Votre compte
Disponibilité par région	Consultez points de terminaison de service.	USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Californie du Nord), USA Ouest (Oregon), Asie-Pacifique (Séoul), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande), Europe (Londres), Moyen-Orient (Bahreïn) et Amérique du Sud (São Paulo)

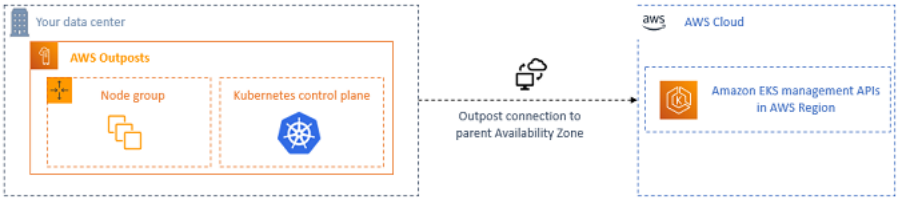
Fonctionnalité	Cluster étendu	Cluster local
Versions Kubernetes mineures	Versions d'Amazon EKS prises en charge.	Versions d'Amazon EKS prises en charge.
Versions de plateforme	Consultez Versions de la plateforme Amazon EKS	Consultez Versions de plateforme de clusters locaux Amazon EKS
Facteurs de forme Outpost	Racks Outpost	Racks Outpost
Interfaces utilisateur	AWS Management Console, AWS CLI, API Amazon EKS, `eksctl`, AWS CloudFormation et Terraform	AWS Management Console, AWS CLI, API Amazon EKS, `eksctl`, AWS CloudFormation et Terraform
Politiques gérées	AmazonEKSClusterPolicy et AmazonEKSServiceRolePolicy	AmazonEKSLocalOutpostClusterPolicy et AmazonEKSLocalOutpostServiceRolePolicy
VPC et sous-réseaux de cluster	Consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux	Consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux
Accès au point de terminaison de cluster	Public, privé ou les deux	Privés uniquement
Authentification du serveur d'API Kubernetes	AWS Identity and Access Management (IAM) et OIDC	IAM et certificats `x.509`
Types de nœud	Autogérés uniquement	Autogérés uniquement
Types de nœuds de calcul	Amazon EC2 à la demande	Amazon EC2 à la demande
Types de stockage de nœuds	SSD `gp2` Amazon EBS et NVMe local	SSD `gp2` Amazon EBS et NVMe local

Fonctionnalité	Cluster étendu	Cluster local
AMI optimisées pour Amazon EKS	Amazon Linux, Windows et Bottlerocket	Amazon Linux uniquement
Versions d'adresses IP	`IPv4` uniquement	`IPv4` uniquement
Modules complémentaires	Modules complémentaires ou autogérés Amazon EKS	Module complémentaire autogéré uniquement
Interface réseau de conteneur s par défaut	Amazon VPC CNI plugin for Kubernetes	Amazon VPC CNI plugin for Kubernetes
Journaux du plan de contrôle Kubernetes	Amazon CloudWatch Logs	Amazon CloudWatch Logs
Equilibrage de charge	Utilisez le AWS Load Balancer Controller pour allouer des Application Load Balancers uniquement (pas de Network Load Balancers)	Utilisez le AWS Load Balancer Controller pour allouer des Application Load Balancers uniquement (pas de Network Load Balancers)
Chiffrement d'enveloppe des secrets	Consultez Activation du chiffrement du secret sur un cluster existant	Non pris en charge
Rôles IAM pour les comptes de service	Consultez Rôles IAM pour les comptes de service	Non pris en charge
Résolution des problèmes	Consultez Dépannage d'Amazon EKS	Consultez Résolution des problèmes de clusters locaux pour Amazon EKS sur AWS Outposts

Clusters locaux pour Amazon EKS sur AWS Outposts

Vous pouvez utiliser des clusters locaux pour exécuter l'intégralité de votre cluster Amazon EKS localement sur AWS Outposts. Cela permet de réduire le risque d'interruption des applications qui

pourrait résulter de déconnexions temporaires du réseau au cloud. Ces déconnexions peuvent être causées par des coupures de fibre ou des événements météorologiques. Étant donné que l'ensemble du cluster Kubernetes s'exécute localement sur les Outposts, les applications restent disponibles. Vous pouvez effectuer des opérations de cluster en cas de déconnexion du réseau au cloud. Pour de plus amples informations, veuillez consulter Préparation aux déconnexions du réseau. Le schéma suivant illustre un déploiement de cluster local.



Les clusters locaux sont généralement disponibles pour une utilisation avec les racks Outposts.

Régions AWS prises en charge

Vous pouvez créer des clusters locaux dans les Régions AWS suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Californie du Nord), USA Ouest (Oregon), Asie-Pacifique (Séoul), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande), Europe (Londres), Moyen-Orient (Bahreïn) et Amérique du Sud (São Paulo). Pour des informations détaillées sur les fonctions prises en charge, consultez Comparaison des options de déploiement.

Rubriques

- Création d'un cluster sur un Outpost
- Versions de plateforme de clusters locaux Amazon EKS
- Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux
- Préparation aux déconnexions du réseau
- Considérations relatives à la capacité
- Résolution des problèmes de clusters locaux pour Amazon EKS sur AWS Outposts

Création d'un cluster sur un Outpost

Cette rubrique fournit une vue d'ensemble des éléments à prendre en compte lors de l'exécution d'un cluster local sur un Outpost. Elle donne également des instructions pour déployer un cluster local sur un Outpost.

Considérations

Important

- Ces considérations ne figurent pas dans la documentation Amazon EKS associée. Si d'autres rubriques de la documentation Amazon EKS entrent en conflit avec les considérations présentées ici, suivez ces dernières.
- Ces considérations sont sujettes à modification et peuvent changer fréquemment. Nous vous recommandons donc de consulter régulièrement cette rubrique.
- La plupart des considérations sont différentes de celles relatives à la création d'un cluster sur le AWS Cloud.

- Les clusters locaux prennent uniquement en charge les racks Outpost. Un seul cluster local peut s'exécuter sur plusieurs racks Outpost physiques qui constituent un seul Outpost logique. Un seul cluster local ne peut pas s'exécuter sur plusieurs Outposts logiques. Chaque Outpost logique possède un seul ARN d'Outpost.
- Les clusters locaux exécutent et gèrent le plan de contrôle Kubernetes sur votre compte sur l'Outpost. Vous ne pouvez pas exécuter de charges de travail sur les instances du plan de contrôle Kubernetes ni modifier les composants du plan de contrôle Kubernetes. Ces nœuds sont gérés par le service Amazon EKS. Les modifications apportées au plan de contrôle Kubernetes ne sont pas conservées en cas d'actions de gestion Amazon EKS automatiques, telles que l'application de correctifs.
- Les clusters locaux prennent en charge les modules complémentaires autogérés et les groupes de nœuds Amazon Linux autogérés. Les modules complémentaires Amazon VPC CNI plugin for Kubernetes, `kube-proxy` et CoreDNS sont automatiquement installés sur les clusters locaux.
- Les clusters locaux nécessitent l'utilisation d'Amazon EBS sur les Outposts. Amazon EBS doit être disponible dans votre Outpost pour le stockage du plan de contrôle Kubernetes.
- Les clusters locaux utilisent Amazon EBS sur les Outposts. Amazon EBS doit être disponible dans votre Outpost pour le stockage du plan de contrôle Kubernetes. Les Outposts prennent en charge les volumes gp2 Amazon EBS uniquement.
- Les `PersistentVolumes` Kubernetes basés sur Amazon EBS sont pris en charge à l'aide du pilote CSI Amazon EBS.

Prérequis

- Connaissance des options de déploiement des Outposts, des Considérations relatives à la capacité et des Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux.
- Un Outpost existant. Pour plus d'informations, consultez What is AWS Outposts.
- L'outil de ligne de commande `kubectl` est installé sur votre ordinateur ou AWS CloudShell. La version peut être identique à la version Kubernetes de votre cluster ou être maximum une version mineure antérieure ou ultérieure. Par exemple, si la version de votre cluster est `1.29`, vous pouvez utiliser la version `kubectl` `1.28`, `1.29` ou `1.30`. Pour installer ou mettre à niveau `kubectl`, veuillez consulter Installation ou mise à jour de `kubectl`.
- Version `2.12.3` ou ultérieure ou version `1.27.160` ou ultérieure du AWS Command Line Interface (AWS CLI) installé et configuré sur votre appareil ou AWS CloudShell. Pour vérifier votre version actuelle, utilisez **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Les gestionnaires de package, par exemple `yum`, `apt-get`, Homebrew ou macOS, sont souvent antérieurs de plusieurs versions à la AWS CLI. Pour installer la dernière version, consultez Installation, mise à jour et désinstallation de l' AWS CLI et Configuration rapide avec aws configure dans le Guide de l'utilisateur AWS Command Line Interface . La AWS CLI version installée AWS CloudShell peut également avoir plusieurs versions de retard par rapport à la dernière version. Pour le mettre à jour, consultez la section Installation AWS CLI dans votre répertoire personnel dans le guide de AWS CloudShell l'utilisateur.
- Un principal IAM avec des autorisations pour `create` et `describe` un cluster Amazon EKS. Pour plus d'informations, consultez Créer un cluster Kubernetes local sur un Outpost et Affichage de la liste ou description de tous les clusters.

Lorsqu'un cluster Amazon EKS local est créé, le principal IAM qui crée le cluster est ajouté de manière permanente. Le principal est spécifiquement ajouté à la table d'autorisation RBAC Kubernetes en tant qu'administrateur. Cette entité possède des autorisations `system:masters`. L'identité de cette entité n'est pas visible dans la configuration de votre cluster. Il est donc important de noter l'entité qui a créé le cluster et de veiller à ne jamais la supprimer. Initialement, seul le principal qui a créé le serveur peut effectuer des appels vers le serveur d'API Kubernetes à l'aide de `kubectl`. Si vous utilisez la console pour créer le cluster, assurez-vous que les mêmes informations d'identification IAM figurent dans la chaîne d'identification du AWS SDK lorsque vous exécutez des `kubectl` commandes sur votre cluster. Une fois votre cluster créé, vous pouvez accorder à d'autres principaux IAM l'accès à votre cluster.

Pour créer un cluster Amazon EKS local

Vous pouvez créer un cluster local avec `eksctl`, la AWS Management Console, la AWS CLI, l'API Amazon EKS, les kits SDK AWS, AWS CloudFormation ou Terraform.

1. Créez un cluster local.

   eksctl

   Prérequis

   Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

   Pour créer votre cluster avec **`eksctl`**

   1. Copiez les contenus suivants sur votre appareil. Remplacez les valeurs suivantes, puis exécutez la commande modifiée pour créer le fichier `outpost-control-plane.yaml` :
      - Remplacez *`region-code`* par la Région AWS prise en charge dans laquelle vous souhaitez créer votre cluster.
      - Remplacez *`my-cluster`* par un nom pour votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
      - Remplacez *`vpc-ExampleID1`* et *`subnet-ExampleID1`* par les identifiants de votre VPC et de votre sous-réseau existants. Le VPC et le sous-réseau doivent répondre aux exigences dans Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux.
      - Remplacez *`uniqueid`* par l'identifiant de votre avant-poste.
      - Remplacez *`m5.large`* par un type d'instance disponible sur votre Outpost. Avant de choisir un type d'instance, consultez Considérations relatives à la capacité. Trois instances du plan de contrôle sont déployées. Vous ne pouvez pas modifier ce nombre.

      ```
      cat >outpost-control-plane.yaml <<EOF
      apiVersion: eksctl.io/v1alpha5
      ```

```
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "1.24"

vpc:
  clusterEndpoints:
    privateAccess: true
  id: "vpc-vpc-ExampleID1"
  subnets:
    private:
      outpost-subnet-1:
        id: "subnet-subnet-ExampleID1"

outpost:
  controlPlaneOutpostARN: arn:aws:outposts:region-code:111122223333:outpost/
op-uniqueid
  controlPlaneInstanceType: m5.large
EOF
```

Pour voir la liste complète de toutes les options disponibles et des valeurs par défaut, consultez AWS Outposts Support et Config file schema (langue française non garantie) dans la documentation `eksctl`.

2. Créez le cluster à l'aide du fichier de configuration créé à l'étape précédente. `eksctl` crée un VPC et un sous-réseau sur votre Outpost pour y déployer le cluster.

```
eksctl create cluster -f outpost-control-plane.yaml
```

L'approvisionnement de cluster dure plusieurs minutes. Pendant la création du cluster, plusieurs lignes de sortie apparaissent. La dernière ligne de sortie est similaire à celle de l'exemple suivant.

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> Pour afficher la plupart des options qui peuvent être spécifiées lors de la création d'un cluster avec `eksctl`, utilisez la commande **`eksctl create cluster --help`**. Pour consulter toutes les options disponibles, vous pouvez utiliser un fichier `config`. Pour plus d'informations, consultez Utilisation des fichiers de configuration et du schéma du fichier de configuration dans la documentation `eksctl`. Vous pouvez trouver des exemples de fichiers de configuration sur GitHub.

`Eksctl` a automatiquement créé une entrée d'accès pour le principal IAM (utilisateur ou rôle) qui a créé le cluster et a accordé à l'administrateur principal IAM des autorisations sur les objets Kubernetes du cluster. Si vous ne souhaitez pas que le créateur du cluster dispose d'un accès administrateur sur les objets Kubernetes du cluster, ajoutez le texte suivant au fichier de configuration précédent : **`bootstrapClusterCreatorAdminPermissions: false`** (au même niveau que `metadata`, `vpc` et `outpost`). Si vous avez ajouté cette option, après la création du cluster, vous devez créer une entrée d'accès pour au moins un principal IAM, sinon aucun principal IAM n'aura accès aux objets Kubernetes du cluster.

AWS Management Console

Prérequis

Un VPC et un sous-réseau existants qui répondent aux exigences Amazon EKS. Pour plus d'informations, consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux.

Pour créer votre cluster à l'aide du AWS Management Console

1. Si vous avez déjà un rôle IAM de cluster local, ou si vous allez créer votre cluster avec `eksctl`, vous pouvez ignorer cette étape. Par défaut, `eksctl` crée un rôle pour vous.
   a. Exécutez la commande suivante pour créer un fichier JSON de politique d'approbation IAM.

```
cat >eks-local-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

b. Créez le rôle IAM de cluster Amazon EKS. Pour créer un rôle IAM, le principal IAM qui crée le rôle doit se voir attribuer l'action (autorisation) IAM `iam:CreateRole`.

```
aws iam create-role --role-name myAmazonEKSLocalClusterRole --assume-role-
policy-document file://"eks-local-cluster-role-trust-policy.json"
```

c. Attachez la politique gérée par Amazon EKS appelée `AmazonEKSLocalOutpostClusterPolicy` au rôle. Pour attacher une politique IAM à un principal IAM, le principal qui attache la politique doit se voir attribuer l'une des actions (autorisations) IAM `iam:AttachUserPolicy` ou `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSLocalOutpostClusterPolicy --role-name myAmazonEKSLocalClusterRole
```

2. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
3. En haut de l'écran de la console, assurez-vous que vous avez sélectionné une Région AWS prise en charge.
4. Choisissez Add cluster (Ajouter un cluster), puis choisissez Create (Créer).
5. Sur la page Configurer le cluster, saisissez ou sélectionnez les valeurs des champs suivants :
   - Emplacement du plan de contrôle Kubernetes : choisissez AWS Outposts.
   - ID Outpost (ID d'Outpost) – Choisissez l'ID de l'Outpost sur lequel vous souhaitez créer votre plan de contrôle.
   - Instance type (Type d'instance) – Sélectionnez un type d'instance. Seuls les types d'instances disponibles dans votre Outpost sont affichés. Dans la liste déroulante, chaque type d'instance indique le nombre de nœuds pour lesquels le type d'instance est recommandé. Avant de choisir un type d'instance, consultez Considérations relatives à

la capacité. Tous les réplicas sont déployés à l'aide du même type d'instance. Vous ne pouvez pas modifier le type d'instance une fois le cluster créé. Trois instances du plan de contrôle sont déployées. Vous ne pouvez pas modifier ce nombre.

- Nom : nom de votre cluster. Il doit être unique dans votre Compte AWS. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.
- Version de Kubernetes : choisissez la version de Kubernetes que vous souhaitez utiliser pour votre cluster. Nous vous recommandons de sélectionner la dernière version, sauf si vous devez utiliser une version antérieure.
- Rôle de service de cluster : choisissez le rôle IAM du cluster Amazon EKS que vous avez créé à l'étape précédente pour permettre au plan de Kubernetes contrôle de gérer les AWS ressources.
- Accès administrateur du clusterKubernetes : si vous souhaitez que le principal IAM (rôle ou utilisateur) qui crée le cluster dispose d'un accès administrateur aux objets Kubernetes du cluster, acceptez la valeur par défaut (autoriser). Amazon EKS crée une entrée d'accès pour le principal IAM et accorde des autorisations d'administrateur du cluster à cette entrée d'accès. Pour plus d'informations sur les entrées d'accès, consultez Gérer les entrées d'accès.

  Si vous souhaitez qu'un principal IAM différent du principal qui crée le cluster dispose d'un accès administrateur aux objets du cluster Kubernetes, choisissez l'option Interdire. Après la création du cluster, tout principal IAM disposant des autorisations IAM pour créer des entrées d'accès peut ajouter des entrées d'accès pour tous les principaux IAM ayant besoin d'accéder aux objets du cluster Kubernetes. Pour plus d'informations sur les autorisations IAM requises, consultez la rubrique Actions définies par Amazon Elastic Kubernetes Service dans la Référence des autorisations de service. Si vous choisissez l'option d'interdiction et que vous ne créez aucune entrée d'accès, aucun principal IAM n'aura accès aux objets Kubernetes du cluster.
- Identifications◦: (facultatif) ajoutez des identifications à votre cluster. Pour plus d'informations, consultez Étiquetage de vos ressources Amazon EKS.

Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.

6. Sur la page Spécifier les réseaux sélectionnez des valeurs pour les champs suivants :

- VPC – Choisissez un VPC existant. Le VPC doit disposer d'un nombre suffisant d'adresses IP disponibles pour le cluster, pour tous les nœuds et pour les autres ressources Kubernetes que vous souhaitez créer. Votre VPC doit répondre aux exigences indiquées dans Exigences et considérations requises pour le VPC.
- Sous-réseaux : par défaut, tous les sous-réseaux disponibles dans le VPC spécifié dans le champ précédent sont présélectionnés. Les sous-réseaux que vous choisissez doivent respecter les exigences indiquées dans Exigences et considérations requises pour les sous-réseaux.

  Groupes de sécurité : (facultatif) spécifiez un ou plusieurs groupes de sécurité créant des interfaces réseau auxquelles vous souhaitez associer Amazon EKS. Amazon EKS crée automatiquement un groupe de sécurité qui permet la communication entre votre cluster et votre VPC. Amazon EKS associe ce groupe de sécurité, et tout ce que vous choisissez, aux interfaces réseau qu'il crée. Pour plus d'informations sur le groupe de sécurité de cluster créé par Amazon EKS, consultez Considérations et exigences relatives aux groupes de sécurité Amazon EKS. Vous pouvez modifier les règles dans le groupe de sécurité de cluster créé par Amazon EKS. Si vous choisissez d'ajouter vos propres groupes de sécurité, vous ne pouvez pas modifier ceux que vous choisissez après la création du cluster. Pour que les hôtes sur site puissent communiquer avec le point de terminaison du cluster, vous devez autoriser le trafic entrant provenant du groupe de sécurité du cluster. Pour les clusters qui n'ont pas de connexion Internet entrante et sortante (également connus sous le nom de clusters privés), vous devez effectuer l'une des opérations suivantes :

  - Ajoutez le groupe de sécurité associé aux points de terminaison d'un VPC requis. Pour plus d'informations sur les points de terminaison requis, consultez Points de terminaison d'un VPC d'interface (français non garanti) dans Accès au sous-réseau à Services AWS.
  - Modifiez le groupe de sécurité qu'Amazon EKS a créé pour autoriser le trafic provenant du groupe de sécurité associé aux points de terminaison d'un VPC.

Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.

7. Sur la page Configurer l'observabilité, vous pouvez éventuellement choisir les options de Métriques et de Journalisation du plan de contrôle que vous voulez activer. Par défaut, chaque type de journal est désactivé.
   - Pour plus d'informations sur l'option de métriques Prometheus, consultez Activation des métriques Prometheus lors de la création d'un cluster.

- Pour plus d'informations sur les options de Journalisation du plan de contrôle, consultez Journalisation de plan de contrôle d'Amazon EKS.

Lorsque vous avez terminé d'utiliser cette page, choisissez Suivant.

8. Sur la page Vérifier et créer, passez en revue les informations que vous avez saisies ou sélectionnées sur les pages précédentes. Si vous devez apporter des modifications, choisissez Modifier. Quand vous êtes satisfait, choisissez Créer. Le champ État affiche EN COURS DE CRÉATION pendant que le cluster est provisionné.

   L'approvisionnement de cluster dure plusieurs minutes.

2. Une fois votre cluster créé, vous pouvez afficher les instances du plan de contrôle Amazon EC2 qui ont été créées.

```
aws ec2 describe-instances --query 'Reservations[*].Instances[*].{Name:Tags[?
Key==`Name`]|[0].Value}' | grep my-cluster-control-plane
```

   L'exemple qui suit illustre un résultat.

```
"Name": "my-cluster-control-plane-id1"
"Name": "my-cluster-control-plane-id2"
"Name": "my-cluster-control-plane-id3"
```

   Chaque instance est rejetée avec `node-role.eks-local.amazonaws.com/control-plane` pour qu'aucune charge de travail ne soit planifiée sur les instances du plan de contrôle. Pour plus d'informations sur les rejets, consultez Taints and Tolerations (langue française non garantie) dans la documentation Kubernetes. Amazon EKS surveille en permanence l'état des clusters locaux. Nous effectuons des actions de gestion automatiques, telles que des correctifs de sécurité et la réparation des instances défectueuses. Lorsque des clusters locaux sont déconnectés du cloud, nous prenons des mesures pour nous assurer que le cluster retrouve un état sain lors de la reconnexion.

3. Si vous avez créé votre cluster à l'aide de `eksctl`, vous pouvez sauter cette étape. `eksctl` complète cette étape pour vous. Activez `kubectl` pour communiquer avec votre cluster en ajoutant un nouveau contexte au fichier `kubectl config`. Pour savoir comment créer et mettre à jour le fichier, consultez Création ou mise à jour d'un fichier `kubeconfig` pour un cluster Amazon EKS.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

L'exemple qui suit illustre un résultat.

```
Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/username/.kube/config
```

4. Pour vous connecter au serveur d'API Kubernetes de votre cluster local, vous devez avoir accès à la passerelle locale du sous-réseau ou vous connecter depuis le VPC. Pour plus d'informations sur la connexion d'un rack Outpost à votre réseau local, consultez la section Fonctionnement des passerelles locales pour les racks dans le Guide de l'utilisateur. AWS Outposts Si vous utilisez le routage VPC direct et que le sous-réseau Outpost possède un acheminement vers votre passerelle locale, les adresses IP privées des instances du plan de contrôle Kubernetes sont automatiquement diffusées sur votre réseau local. Le point de terminaison du serveur d'API Kubernetes du cluster local est hébergé dans Amazon Route 53 (Route 53). Le point de terminaison du service d'API peut être résolu par des serveurs DNS publics vers les adresses IP privées des serveurs d'API Kubernetes.

   Les instances de plan de contrôle Kubernetes des clusters locaux sont configurées avec des interfaces réseau Elastic statiques dotées d'adresses IP privées fixes qui restent les mêmes tout au long du cycle de vie du cluster. Les appareils qui interagissent avec le serveur d'API Kubernetes peuvent ne pas avoir de connectivité à Route 53 pendant les déconnexions du réseau. Si tel est le cas, nous recommandons de configurer `/etc/hosts` avec les adresses IP privées statiques pour la poursuite des opérations. Nous vous recommandons également de configurer des serveurs DNS locaux et de les connecter à votre Outpost. Pour en savoir plus, consultez la documentation AWS Outposts. Exécutez la commande suivante pour confirmer que la communication est établie avec votre cluster.

   ```
   kubectl get svc
   ```

   L'exemple qui suit illustre un résultat.

   ```
   NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
   kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
   ```

5. (Facultatif) Testez l'authentification auprès de votre cluster local lorsqu'il est dans un état déconnecté de l' AWS Cloud. Pour obtenir des instructions, veuillez consulter Préparation aux déconnexions du réseau.

Ressources internes

Amazon EKS crée les ressources suivantes sur votre cluster. Les ressources sont destinées à un usage interne d'Amazon EKS. Pour le bon fonctionnement de votre cluster, ne modifiez pas ces ressources.

- Les Pods miroir suivants :
  - `aws-iam-authenticator-`*`node-hostname`*
  - `eks-certificates-controller-`*`node-hostname`*
  - `etcd-`*`node-hostname`*
  - `kube-apiserver-`*`node-hostname`*
  - `kube-controller-manager-`*`node-hostname`*
  - `kube-scheduler-`*`node-hostname`*
- Les modules complémentaires autogérés suivants :
  - `kube-system/coredns`
  - `kube-system/kube-proxy` (non créé tant que vous n'avez pas ajouté votre premier nœud)
  - `kube-system/aws-node` (non créé tant que vous n'avez pas ajouté votre premier nœud). Les clusters locaux utilisent le plugin Amazon VPC CNI plugin for Kubernetes pour les réseaux de clusters Ne modifiez pas la configuration des instances du plan de contrôle (pods nommés `aws-node-controlplane-*`). Il existe des variables de configuration que vous pouvez utiliser pour modifier la valeur par défaut lorsque le plugin crée de nouvelles interfaces réseau. Pour plus d'informations, consultez la documentation sur GitHub.
- Les services suivants :
  - `default/kubernetes`
  - `kube-system/kube-dns`
- Une politique `PodSecurityPolicy` nommée `eks.system`
- Un rôle `ClusterRole` nommé `eks:system:podsecuritypolicy`
- Un rôle `ClusterRoleBinding` nommé `eks:system`
- Une PodSecuritypolitique par défaut
- Outre le groupe de sécurité du cluster, Amazon EKS crée un groupe de sécurité nommé dans votre Compte AWS répertoire`eks-local-internal-do-not-use-or-edit-`*`cluster-name`*`-`*`uniqueid`*. Ce groupe de sécurité permet au trafic de circuler librement entre les composants Kubernetes exécutés sur les instances du plan de contrôle.

Étapes suivantes recommandées :

- Accordez au principal IAM qui a créé le cluster les autorisations requises pour consulter les Kubernetes ressources dans AWS Management Console
- Accordez aux entités IAM l'accès à votre cluster. Si vous souhaitez que les entités affichent les ressources Kubernetes dans la console Amazon EKS, accordez les Autorisations nécessaires aux entités.
- Configurer la journalisation de votre cluster
- Familiarisez-vous avec ce qui se passe en cas de déconnexion du réseau.
- Ajouter des nœuds à votre cluster
- Envisagez de mettre en place un plan de sauvegarde pour votre `etcd`. Amazon EKS ne prend pas en charge la sauvegarde et la restauration automatisées d'`etcd` pour les clusters locaux. Pour plus d'informations, consultez Sauvegarder un cluster `etcd` dans la documentation Kubernetes. Les deux options principales utilisent `etcdctl` pour automatiser la prise d'instantanés ou l'utilisation de la sauvegarde des volumes de stockage Amazon EBS.

Versions de plateforme de clusters locaux Amazon EKS

Les versions de la plateforme de cluster local représentent les capacités du cluster Amazon EKS sur AWS Outposts. Les versions comprennent les composants qui s'exécutent sur le plan de contrôle Kubernetes, pour lesquels les drapeaux du serveur d'API Kubernetes sont activés. Elles comprennent également la version actuelle du correctif Kubernetes. Une ou plusieurs versions de plateforme sont associées à chaque version mineure de Kubernetes. Les versions de plateforme pour les différentes versions mineures de Kubernetes sont indépendantes. Les versions de plateforme pour les clusters locaux et les clusters Amazon EKS dans le cloud sont indépendantes.

Lorsqu'une nouvelle version mineure de Kubernetes est disponible pour les clusters locaux, comme la version `1.28`, la version initiale de plateforme pour cette version mineure de Kubernetes commence à `eks-local-outposts.1`. Cependant, Amazon EKS publie régulièrement de nouvelles versions de plateforme pour activer de nouveaux paramètres de plan de contrôle Kubernetes et fournir des correctifs de sécurité.

Lorsque de nouvelles versions de plateforme de clusters locaux deviennent disponibles pour une version mineure :

- Le numéro de version de plateforme est incrémenté (`eks-local-outposts.n+1`).

- Amazon EKS met automatiquement à jour tous les clusters locaux existants vers la dernière version de plateforme pour les versions mineures Kubernetes correspondantes. Les mises à jour automatiques de versions de plateforme existantes sont déployées de façon incrémentielle. Le processus de déploiement peut prendre du temps. Si vous avez besoin immédiatement des fonctions de la dernière version de plateforme, nous vous recommandons de créer un nouveau cluster local.
- Amazon EKS peut publier une nouvelle AMI de nœud avec une version de correctif correspondante. Toutes les versions correctives sont compatibles entre le plan de contrôle Kubernetes et les AMI de nœud pour une seule version mineure de Kubernetes.

Les nouvelles versions de plateforme n'introduisent pas des modifications importantes ou ne provoquent pas des interruptions de service.

Les clusters locaux sont toujours créés avec la dernière version de plateforme disponible (`eks-local-outposts.n`) pour la version Kubernetes spécifiée.

Les versions actuelles et récentes de plateforme sont décrites dans les tableaux ci-dessous.

Kubernetes version **`1.28`**

Les contrôleurs d'admission suivants sont activés pour toutes les versions de plateforme `1.28` : `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `Priority`, `PodSecurity`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy` et `ValidatingAdmissionWebhook`.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
`1.28.6`	`eks-local-outposts.5`	Mise à jour de la version v1.19.3 de Bottlerocket contenant les dernières corrections de bogues	18 avril 2024

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
		pour prendre en charge le démarrage local dans Outposts.	
1.28.6	eks-local-outposts.4	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations. Support rétabli ou démarrage local dans Outposts. BottlerocketVersion rétrogradée à pour des v1.15.1 raisons de compatibilité.	2 avril 2024
1.28.6	eks-local-outposts.3	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	22 mars 2024
1.28.6	eks-local-outposts.2	Nouvelle version de la plateforme avec des correctifs de sécurité et des améliorations vers lesquels kube-proxy a été mis à jour. v1.28.6 AWS Authentificateur IAM mis à jour vers. v0.6.17 Le plug-in Amazon VPC CNI pour Kubernetes a été rétrogradé à pour des raisons de compatibilité. v1.13.2 BottlerocketVersion mise à jour versv1.19.2.	8 mars 2024
1.28.1	eks-local-outposts.1	Publication initiale de la version v1.28 Kubernetes pour les clusters Amazon EKS locaux sur Outpost	4 octobre 2023

Kubernetes version **1.27**

Les contrôleurs d'admission suivants sont activés pour toutes les versions de plateforme `1.27` : `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `Priority`, `PodSecurity`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy` et `ValidatingAdmissionWebhook`.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
`1.27.10`	`eks-local-outposts.5`	Nouvelle plateforme avec correctifs et améliorations de sécurité.	2 avril 2024
`1.27.10`	`eks-local-outposts.4`	Nouvelle plate-forme avec des correctifs de sécurité et des améliorations. Kube-prox y mis à jour vers. `v1.27.10` AWS Authentificateur IAM mis à jour vers. `v0.6.17` BottlerocketVersion mise à jour vers`v1.19.2`.	22 mars 2024
`1.27.3`	`eks-local-outposts.3`	Nouvelle version de plateform e avec des corrections de sécurité et des améliorations. Mise à jour de `kube-proxy` vers `v1.27.3`. Mise à jour du plugin CNI Amazon VPC pour Kubernetes vers `v1.13.2`.	14 juillet 2023

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.27.1	eks-local-outposts.2	Mise à jour de l'image de CoreDNS vers v1.10.1	22 juin 2023
1.27.1	eks-local-outposts.1	Publication initiale de la version Kubernetes pour les clusters Amazon EKS 1.27 locaux sur Outposts.	30 mai 2023

Kubernetes version **1.26**

Les contrôleurs d'admission suivants sont activés pour toutes les versions de plateforme 1.26 : CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, Priority, PodSecurity, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition, ValidatingAdmissionPolicy et ValidatingAdmissionWebhook.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.26.13	eks-local-outposts.5	Nouvelle version de la plateform e avec correctifs et améliorat ions de sécurité. Kube-prox y mis à jour vers. v1.26.13 AWS Authentificateur IAM mis à jour vers. v0.6.17 BottlerocketVersion mise à jour versv1.19.2.	22 mars 2024

Kubernetes version **1.25**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.25 :CertificateApproval,CertificateSigning,CertificateSubjectRestriction,DefaultIngressClass,DefaultStorageClass,DefaultTolerationSeconds,LimitRanger,MutatingAdmissionWebhook,NamespaceLifecycle,NodeRestriction,PersistentVolumeClaimResize,Priority,PodSecurity,ResourceQuota,RuntimeClass,ServiceAccount,StorageObjectInUseProtection,ValidatingAdmissionPolicy,ValidatingAdmissionWebhook.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.25.16	eks-local-outposts.7	Nouvelle version de la plateforme avec correctifs et améliorations de sécurité. Kube-proxy mis à jour vers. v1.25.16 AWS Authentificateur IAM mis à jour vers. v0.6.17 BottlerocketVersion mise à jour versv1.19.2.	22 mars 2024
1.25.11	eks-local-outposts.6	Nouvelle version de plateforme avec des corrections de sécurité et des améliorations. Mise à jour de kube-proxy vers v1.25.11. Mise à jour du plugin CNI Amazon VPC pour Kubernetes vers v1.13.2.	14 juillet 2023
1.25.9	eks-local-outposts.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	13 juillet 2023
1.25.6	eks-local-outposts.4	Version de Bottlerocket mise à jour vers 1.13.2	2 mai 2023
1.25.6	eks-local-outposts.3	Le système d'exploitation de l'instance du plan de contrôle Amazon EKS a été mis à jour vers la version Bottlerocket et le	14 avril 2023

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
		plug-in v1.13.1 Amazon VPC CNI pour Kubernetes a été mis à jour vers la version. v1.12.6	
1.25.6	eks-local-outposts.2	Collecte de diagnostics améliorée pour les instances du plan de contrôle Kubernetes.	8 mars 2023
1.25.6	eks-local-outposts.1	Publication initiale de la version Kubernetes pour les clusters Amazon EKS 1.25 locaux sur Outposts.	1 mars 2023

Kubernetes version **1.24**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version 1.24 :DefaultStorageClass,DefaultTolerationSeconds,LimitRanger,MutatingAdmissetDefaultIngressClass.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.24.17	eks-local-outposts.7	Nouvelle version de la plateforme avec correctifs et améliorations de sécurité. Kube-proxy mis à jour vers. v1.25.16 AWS Authentif icateur IAM mis à jour. v0.6.17 BottlerocketVersion mise à jour versv1.19.2.	22 mars 2024

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
1.24.15	eks-local-outposts.6	Nouvelle version de plateforme avec des corrections de sécurité et des améliorations. Mise à jour de kube-proxy vers v1.24.15. Mise à jour du plugin CNI Amazon VPC pour Kubernetes vers v1.13.2.	14 juillet 2023
1.24.13	eks-local-outposts.5	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	13 juillet 2023
1.24.9	eks-local-outposts.4	Version de Bottlerocket mise à jour vers 1.13.2	2 mai 2023
1.24.9	eks-local-outposts.3	Le système d'exploitation de l'instance du plan de contrôle Amazon EKS a été mis à jour vers la version Bottlerocket et le plug-in v1.13.1 Amazon VPC CNI pour Kubernetes a été mis à jour vers la version. v1.12.6	14 avril 2023
1.24.9	eks-local-outposts.2	Collecte de diagnostics améliorée pour les instances du plan de contrôle Kubernetes.	8 mars 2023
1.24.9	eks-local-outposts.1	Publication initiale de la version Kubernetes pour les clusters Amazon EKS 1.24 locaux sur Outposts.	17 janvier 2023

Kubernetes version **1.23**

Les contrôleurs d'admission suivants sont activés pour toutes les plateformes version `1.23` :`DefaultStorageClass`,`DefaultTolerationSeconds`,`LimitRanger`,`MutatingAdmissetDefaultIngressClass`.

Version de Kubernetes	Version de la plateforme Amazon EKS	Notes de mise à jour	Date de publication
`1.23.17`	`eks-local-outposts.6`	Nouvelle version de la plateforme avec des corrections de sécurité et des améliorations.	13 juillet 2023
`1.23.17`	`eks-local-outposts.5`	Nouvelle version de la plateforme avec correctifs et améliorations de sécurité. Kube-proxy mis à jour vers. `v1.23.17` BottlerocketVersion mise à jour vers`v1.14.1`.	6 juillet 2023
`1.23.15`	`eks-local-outposts.4`	Le système d'exploitation de l'instance du plan de contrôle Amazon EKS a été mis à jour vers la version Bottlerocket et le plug-in `v1.13.1` Amazon VPC CNI pour Kubernetes a été mis à jour vers la version. `v1.12.6`	14 avril 2023
`1.23.15`	`eks-local-outposts.3`	Collecte de diagnostics améliorée pour les instances du plan de contrôle Kubernetes.	8 mars 2023
`1.23.15`	`eks-local-outposts.2`	Publication initiale de la version Kubernetes pour les clusters Amazon EKS `1.23` locaux sur Outposts.	17 janvier 2023

Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux

Lorsque vous créez un cluster local, vous spécifiez un VPC et au moins un sous-réseau privé qui fonctionne sur les Outposts. Cette rubrique fournit une vue d'ensemble des exigences et considérations spécifiques au VPC et aux sous-réseaux pour votre cluster local.

Exigences et considérations requises pour le VPC

Lorsque vous créez un cluster local, le VPC que vous spécifiez doit répondre aux exigences et aux considérations suivantes :

- Assurez-vous que le VPC dispose de suffisamment d'adresses IP pour le cluster local, les nœuds et les autres ressources Kubernetes que vous voulez créer. Si le VPC que vous souhaitez utiliser ne dispose pas de suffisamment d'adresses IP, augmentez le nombre d'adresses IP disponibles. Vous pouvez effectuer cette opération en associant des blocs d'adresse CIDR (Routage inter-domaines sans classe) supplémentaires avec votre VPC. Vous pouvez associer des blocs d'adresses CIDR privés (RFC 1918) et publics (non-RFC 1918) à votre VPC avant ou après la création de votre cluster. Un cluster peut prendre jusqu'à 5 heures avant qu'un bloc d'adresse CIDR associé à un VPC ne soit reconnu.
- Le VPC ne peut pas avoir de préfixes IP ou de blocs d'adresses CIDR IPv6 attribués. En raison de ces contraintes, les informations couvertes dans Augmenter le nombre d'adresses IP disponibles pour vos nœuds Amazon EC2 et `IPv6`adresses pour les clustersPods, et services ne sont pas applicables à votre VPC.
- Le VPC possède un nom d'hôte DNS et une résolution DNS activés. Sans ces fonctions, la création du cluster local échoue, et vous devez activer les fonctions et recréer votre cluster. Pour plus d'informations, consultez Attributs DNS pour votre VPC dans le guide de l'utilisateur d'Amazon VPC.
- Pour accéder à votre cluster local via votre réseau local, le VPC doit être associé à la table de routage de la passerelle locale de votre Outpost. Pour de plus amples informations, veuillez consulter Associations de VPC dans le Guide de l'utilisateur AWS Outposts.

Exigences et considérations requises pour les sous-réseaux

Lorsque vous créez le cluster, spécifiez au moins un sous-réseau privé. Si vous spécifiez plusieurs sous-réseaux, les instances du plan de contrôle Kubernetes sont réparties uniformément à travers les sous-réseaux. Si plus d'un sous-réseau est spécifié, les sous-réseaux doivent exister sur le

même Outpost. En outre, les sous-réseaux doivent également disposer d'un acheminement et d'autorisations de groupe de sécurité appropriés pour communiquer entre eux. Lorsque vous créez un cluster local, les sous-réseaux que vous spécifiez doivent répondre aux exigences suivantes :

- Les sous-réseaux doivent se trouver sur le même Outpost logique.
- Les sous-réseaux doivent disposer ensemble d'au moins trois adresses IP disponibles pour les instances du plan de contrôle Kubernetes. Si trois sous-réseaux sont spécifiés, chaque sous-réseau doit avoir au moins une adresse IP disponible. Si deux sous-réseaux sont spécifiés, chaque sous-réseau doit avoir au moins deux adresses IP disponibles. Si un sous-réseau est spécifié, le sous-réseau doit avoir au moins trois adresses IP disponibles.
- Les sous-réseaux doivent inclure un acheminement vers la passerelle locale du rack Outpost pour accéder au serveur d'API Kubernetes sur votre réseau local. Si les sous-réseaux n'ont pas d'acheminement vers la passerelle locale du rack Outpost, vous devez communiquer avec votre serveur d'API Kubernetes depuis le VPC.
- Les sous-réseaux doivent utiliser une dénomination basée sur une adresse IP. La dénomination basée sur la ressource Amazon EC2 n'est pas prise en charge par Amazon EKS.

Accès au sous-réseau à Services AWS

Les sous-réseaux privés du cluster local sur Outposts doivent être en mesure de communiquer avec les Services AWS régionaux. Vous pouvez y parvenir en utilisant une passerelle NAT pour l'accès Internet sortant ou, si vous voulez préserver la confidentialité de tout le trafic au sein de votre VPC, en utilisant des points de terminaison d'un VPC d'interface.

Utiliser une passerelle NAT

Les sous-réseaux privés du cluster local sur Outposts doivent inclure une table de routage associée contenant un acheminement vers une passerelle NAT dans un sous-réseau public qui est dans la zone de disponibilité parente de l'Outpost. Le sous-réseau public doit disposer d'un acheminement vers une passerelle Internet. La passerelle NAT permet l'accès à Internet sortant et empêche les connexions entrantes non sollicitées depuis Internet vers les instances de l'Outpost.

Utilisation des points de terminaison de VPC d'interface

Si les sous-réseaux privés du cluster local sur Outposts n'ont pas d'accès Internet sortant ou si vous voulez préserver la confidentialité de tout le trafic au sein de votre VPC, vous devez créer les points de terminaison d'un VPC d'interface et le point de terminaison de passerelle suivants dans un sous-réseau régional avant de créer votre cluster.

Point de terminaison	Type de point de terminaison
com.amazonaws. *region-code* .ssm	utilisateur
com.amazonaws. *region-code* .ssmmessages	utilisateur
com.amazonaws. *region-code* .ec2messages	utilisateur
com.amazonaws. *region-code* .ec2	utilisateur
com.amazonaws. *region-code* .secretsmanager	utilisateur
com.amazonaws. *region-code* .logs	utilisateur
com.amazonaws. *region-code* .sts	utilisateur
com.amazonaws. *region-code* .ecr.api	utilisateur
com.amazonaws. *region-code* .ecr.dkr	utilisateur
com.amazonaws. *region-code* .s3	Passerelle

Les points de terminaison doivent répondre aux critères suivants :

- Créé dans un sous-réseau privé situé dans la zone de disponibilité parent de votre Outpost
- Noms DNS privés activés
- Avoir un groupe de sécurité attaché qui autorise le trafic HTTPS entrant à partir de la plage CIDR du sous-réseau privé d'Outpost.

La création de points de terminaison entraîne des frais. Pour en savoir plus, consultez PricingAWS PrivateLink (Tarification). Si vos Pods doivent accéder à d'autres Services AWS, vous devez créer des points de terminaison supplémentaires. Pour une liste complète des points de terminaison, consultez Les Services AWS qui s'intègrent à AWS PrivateLink.

Création d'un VPC

Vous pouvez créer un VPC qui répond aux exigences précédentes en utilisant l'un des modèles AWS CloudFormation suivants :

- Modèle 1 – Ce modèle crée un VPC avec un sous-réseau privé sur l'Outpost et un sous-réseau public dans l'Région AWS. Le sous-réseau privé dispose d'un acheminement vers Internet via une passerelle NAT qui réside dans le sous-réseau public dans l'Région AWS. Ce modèle peut être utilisé pour créer un cluster local dans un sous-réseau avec un accès Internet de sortie.
- Modèle 2 – Ce modèle crée un VPC avec un sous-réseau privé sur l'Outpost et l'ensemble minimum de points de terminaison d'un VPC requis pour créer un cluster local dans un sous-réseau qui n'a pas d'accès Internet d'entrée ou de sortie (également appelé sous-réseau privé).

Préparation aux déconnexions du réseau

Si votre réseau local a perdu la connectivité avec AWS Cloud, vous pouvez continuer à utiliser votre cluster Amazon EKS local sur un Outpost. Cette rubrique explique comment préparer votre cluster local pour les déconnexions du réseau et les considérations connexes.

Considérations à prendre en compte pour préparer votre cluster local à une déconnexion du réseau :

- Les clusters locaux garantissent la stabilité et la continuité des opérations en cas de déconnexions réseau temporaires et imprévues. AWS Outposts reste une offre entièrement connectée qui agit comme extension du AWS Cloud dans votre centre de données. En cas de déconnexion du réseau entre votre Outpost et AWS Cloud, nous vous recommandons de tenter de rétablir votre connexion. Pour obtenir des instructions, consultez la liste de contrôle de dépannage du réseau du rack AWS Outposts dans le Guide de l'utilisateur AWS Outposts. Pour plus d'informations sur la résolution des problèmes liés aux clusters locaux, consultez Résolution des problèmes de clusters locaux pour Amazon EKS sur AWS Outposts.
- Les Outposts génèrent une métrique `ConnectedStatus` qui permet de surveiller l'état de connectivité de votre Outpost. Pour plus d'informations, consultez Métriques d'Outpost dans le Guide de l'utilisateur AWS Outposts.
- Les clusters locaux utilisent IAM comme mécanisme d'authentification par défaut à l'aide d'AWS Identity and Access Management Authenticator pour Kubernetes. IAM n'est pas disponible lors des déconnexions du réseau. Les clusters locaux prennent en charge un mécanisme d'authentification à l'aide des certificats `x.509` que vous pouvez utiliser pour vous connecter à votre cluster lors des déconnexions du réseau. Pour plus d'informations sur l'obtention et l'utilisation d'un certificat `x.509`

pour votre cluster, consultez Authentification auprès de votre cluster local lors d'une déconnexion du réseau.

- Si vous ne pouvez pas accéder à Route 53 lors des déconnexions du réseau, pensez à utiliser des serveurs DNS locaux dans votre environnement sur site. Les instances du plan de contrôle Kubernetes utilisent des adresses IP statiques. Vous pouvez configurer les hôtes que vous utilisez pour vous connecter à votre cluster avec le nom d'hôte et les adresses IP du point de terminaison comme alternative à l'utilisation de serveurs DNS locaux. Pour plus d'informations, consultez DNS dans le AWS OutpostsGuide de l'utilisateur d'.
- Si vous prévoyez une augmentation du trafic d'applications lorsque le réseau se déconnecte, vous pouvez allouer de la capacité de calcul de réserve dans votre cluster lorsque vous êtes connecté au cloud. Les instances Amazon EC2 sont incluses dans le prix d'AWS Outposts. Ainsi, l'exécution d'instances de rechange n'a pas d'impact sur votre coût d'utilisation d'AWS.
- Lors des déconnexions du réseau pour permettre les opérations de création, de mise à jour et de dimensionnement des charges de travail, les images de conteneur de votre application doivent être accessibles sur le réseau local et votre cluster doit disposer d'une capacité suffisante. Les clusters locaux n'hébergent pas de registre de conteneurs pour vous. Si les Pods ont déjà été exécutés sur ces nœuds, les images de conteneur sont mises en cache sur les nœuds. Si vous avez l'habitude de télécharger les images de conteneur de votre application depuis Amazon ECR dans le cloud, envisagez d'exécuter un registre ou un cache local. Un cache ou un registre local est utile si vous avez besoin d'opérations de création, de mise à jour et de mise à l'échelle pour les ressources de la charge de travail pendant les déconnexions du réseau.
- Les clusters locaux utilisent Amazon EBS comme classe de stockage par défaut pour les volumes persistants et le pilote CSI Amazon EBS pour gérer le cycle de vie des volumes persistants Amazon EBS. Pendant les déconnexions du réseau, les Pods qui sont sauvegardés par Amazon EBS ne peuvent pas être créés, mis à jour ou mis à l'échelle. En effet, ces opérations nécessitent des appels à l'API Amazon EBS dans le cloud. Si vous déployez des charges de travail avec état sur des clusters locaux et que vous avez besoin d'opérations de création, de mise à jour ou de mise à l'échelle lors des déconnexions du réseau, envisagez d'utiliser un autre mécanisme de stockage.
- Il est impossible de créer ou de supprimer des instantanés Amazon EBS si AWS Outposts n'a pas accès aux API pertinentes de la région AWS (telles que les API pour Amazon EBS ou Amazon S3).
- Lors de l'intégration d'ALB (Ingress) à AWS Certificate Manager (ACM), les certificats sont transmis et stockés dans la mémoire de l'instance AWS Outposts ALB Compute. La terminaison TLS actuelle continuera de fonctionner en cas de déconnexion de la Région AWS. Dans ce contexte, les opérations de mutation (telles que les nouvelles définitions d'entrée, les nouvelles

opérations d'API de certificats basés sur ACM, le dimensionnement du calcul ALB ou la rotation des certificats) échoueront. Pour plus d'informations, consultez Résolution des problèmes liés au renouvellement géré des certificats dans le Guide de l'utilisateur AWS Certificate Manager.

- Les journaux du plan de contrôle Amazon EKS sont mis en cache localement sur les instances du plan de contrôle Kubernetes lors des déconnexions du réseau. Lors de la reconnexion, les journaux sont envoyés à CloudWatch Logs in the parentRégion AWS. Vous pouvez utiliser Prometheus, Grafana ou les solutions partenaires d'Amazon EKS pour surveiller le cluster localement à l'aide du point de terminaison des métriques du serveur d'API Kubernetes ou de Fluent Bit pour les journaux.
- Si vous utilisez l'AWS Load Balancer Controller sur les Outposts pour le trafic d'applications, les Pods existants gérés par l'AWS Load Balancer Controller continuent de recevoir du trafic pendant les déconnexions du réseau. Les nouveaux Pods créés lors de déconnexions du réseau ne reçoivent pas de trafic tant que l'Outpost n'est pas reconnecté au AWS Cloud. Envisagez de définir le nombre de réplicas pour vos applications lorsque vous êtes connecté au AWS Cloud afin de répondre à vos besoins de mise à l'échelle lors des déconnexions du réseau.
- Le plugin Amazon VPC CNI plugin for Kubernetes est en mode Secondary IP par défaut. Il est configuré avec `WARM_ENI_TARGET=1`, qui permet au plugin de conserver « une interface réseau entièrement Elastic » contenant les adresses IP disponibles. Pensez à modifier les valeurs `WARM_ENI_TARGET`, `WARM_IP_TARGET` et `MINIMUM_IP_TARGET` en fonction de vos besoins de mise à l'échelle lors d'un état déconnecté. Pour plus d'informations, consultez le `readme`fichier du plugin sur GitHub. Pour obtenir la liste du nombre maximum de Pods celles prises en charge par chaque type d'instance, consultez le `eni-max-pods.txt`fichier sur GitHub.

Authentification auprès de votre cluster local lors d'une déconnexion du réseau

AWS Identity and Access Management (IAM) n'est pas disponible lors des déconnexions du réseau. Vous ne pouvez pas vous authentifier auprès de votre cluster local à l'aide des informations d'identification IAM lorsque vous êtes déconnecté. Cependant, vous pouvez vous connecter à votre cluster sur votre réseau local à l'aide de certificats `x509` en cas de déconnexion. Vous devez télécharger et stocker un certificat `X509` client à utiliser lors des déconnexions. Dans cette rubrique, vous allez voir comment créer et utiliser le certificat pour vous authentifier auprès de votre cluster lorsqu'il est dans un état déconnecté.

1. Créer une demande de signature de certificat.

    a. Générez une demande de signature de certificat.

```
openssl req -new -newkey rsa:4096 -nodes -days 365 \
    -keyout admin.key -out admin.csr -subj "/CN=admin"
```

b. Créez une demande de signature de certificat dans Kubernetes.

```
BASE64_CSR=$(cat admin.csr | base64 -w 0)
cat << EOF > admin-csr.yaml
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: admin-csr
spec:
  signerName: kubernetes.io/kube-apiserver-client
  request: ${BASE64_CSR}
  usages:
  - client auth
EOF
```

2. Créez une demande de signature de certificat à l'aide de `kubectl`.

```
kubectl create -f admin-csr.yaml
```

3. Vérifiez le statut de la demande de signature de certificat.

```
kubectl get csr admin-csr
```

L'exemple qui suit illustre un résultat.

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Pending
```

Kubernetes a créé la demande de signature de certificat.

4. Approuvez la demande de signature de certificat.

```
kubectl certificate approve admin-csr
```

5. Vérifiez à nouveau l'état de la demande de signature de certificat pour approbation.

```
kubectl get csr admin-csr
```

L'exemple qui suit illustre un résultat.

```
NAME       AGE   REQUESTOR                     CONDITION
admin-csr  11m   kubernetes-admin              Approved
```

6. Récupérez et vérifiez le certificat.

   a. Récupérez le certificat.

   ```
   kubectl get csr admin-csr -o jsonpath='{.status.certificate}' | base64 --decode
    > admin.crt
   ```

   b. Vérifiez le certificat.

   ```
   cat admin.crt
   ```

7. Créez une liaison de rôle de cluster pour un utilisateur `admin`.

```
kubectl create clusterrolebinding admin --clusterrole=cluster-admin \
    --user=admin --group=system:masters
```

8. Générez un kubeconfig spécifique à l'utilisateur pour un état déconnecté.

   Vous pouvez générer un fichier `kubeconfig` à l'aide des certificats `admin` téléchargés. Remplacez *`my-cluster`* et *`apiserver-endpoint`* dans les commandes suivantes.

```
aws eks describe-cluster --name my-cluster \
    --query "cluster.certificateAuthority" \
    --output text | base64 --decode > ca.crt
```

```
kubectl config --kubeconfig admin.kubeconfig set-cluster my-cluster \
    --certificate-authority=ca.crt --server apiserver-endpoint --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-credentials admin \
    --client-certificate=admin.crt --client-key=admin.key --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-context admin@my-cluster \
    --cluster my-cluster --user admin
```

```
kubectl config --kubeconfig admin.kubeconfig use-context admin@my-cluster
```

9. Affichez votre fichier `kubeconfig`.

```
kubectl get nodes --kubeconfig admin.kubeconfig
```

10. Si des services sont déjà en production sur votre Outpost, ignorez cette étape. Si Amazon EKS est le seul service qui s'exécute sur votre Outpost et que ce dernier n'est pas en production, vous pouvez simuler une déconnexion du réseau. Avant de passer en production avec votre cluster local, simulez une déconnexion pour vous assurer que vous pouvez accéder à votre cluster lorsqu'il est déconnecté.

    a. Appliquez les règles de pare-feu sur les appareils réseaux qui connectent votre Outpost à la Région AWS. Cela déconnecte le lien de service de l'Outpost. Vous ne pouvez pas créer de nouvelles instances. Les instances en cours d'exécution perdent leur connectivité à la Région AWS et à Internet.

    b. Vous pouvez tester la connection à votre cluster local à l'aide du certificat `x509` en cas de déconnexion. Assurez-vous de remplacer votre `kubeconfig` par le `admin.kubeconfig` que vous avez créé à une étape précédente. Remplacez *`my-cluster`* par le nom de votre cluster local.

    ```
    kubectl config use-context admin@my-cluster --kubeconfig admin.kubeconfig
    ```

    Si vous remarquez des problèmes avec vos clusters locaux alors qu'ils sont déconnectés, nous vous recommandons d'ouvrir un ticket de support.

Considérations relatives à la capacité

Cette rubrique fournit des conseils sur la sélection du type d'instance de plan de contrôle Kubernetes et sur l'utilisation (facultative) de groupes de placement pour répondre aux exigences de haute disponibilité de votre cluster Amazon EKS local sur un Outpost.

Avant de sélectionner le type d'instance (tel que `m5`, `c5` ou `r5`) à utiliser pour le plan de contrôle Kubernetes de votre cluster local sur des Outposts, vérifiez les types d'instances disponibles dans votre configuration Outpost. Après avoir identifié les types d'instance disponibles, sélectionnez la taille de l'instance (telle que `large`, `xlarge` ou `2xlarge`) en fonction du nombre de nœuds requis

par vos charges de travail. Le tableau suivant fournit des recommandations pour choisir une taille d'instance.

> **Note**
>
> Les tailles d'instance doivent avoir été définies sur vos Outposts. Assurez-vous que vous avez suffisamment de capacité pour trois instances de la taille disponible sur vos Outposts pour la durée de vie de votre cluster local. Pour obtenir la liste des types d' Amazon EC2 instances disponibles, consultez les sections Calcul et stockage de la section Fonctionnalités du AWS Outposts rack.

Nombre de modèles	Taille des instances du plan de contrôle Kubernetes
1-20	`large`
21–100	`xlarge`
101–250	`2xlarge`
251–500	`4xlarge`

Le stockage du plan de contrôle Kubernetes nécessite 246 Go de stockage Amazon EBS pour chaque cluster local afin répondre aux exigences d'IOPS d'`etcd`. Lorsque le cluster local est créé, les volumes Amazon EBS sont alloués automatiquement pour vous.

Placement du plan de contrôle

Lorsque vous ne spécifiez aucun groupe de placement avec la propriété `OutpostConfig.ControlPlanePlacement.GroupName`, les instances Amazon EC2 provisionnées pour votre plan de contrôle Kubernetes ne font l'objet d'aucune contrainte de placement matériel spécifique sur la capacité sous-jacente disponible sur votre Outpost.

Vous pouvez utiliser des groupes de placement pour répondre aux exigences de haute disponibilité de votre cluster Amazon EKS local sur un Outpost. En spécifiant un groupe de placement lors de la création du cluster, vous influencez le placement des instances de plan de contrôle Kubernetes. Les

instances sont réparties sur du matériel sous-jacent indépendant (racks ou hôtes), minimisant ainsi l'impact des instances corrélées en cas de panne matérielle.

Prérequis

Le type de répartition que vous pouvez configurer dépend du nombre de racks Outpost dont vous disposez dans votre déploiement.

- Déploiements avec un ou deux racks physiques au sein d'un seul Outpost logique : au moins trois hôtes doivent être configurés avec le type d'instance que vous choisissez pour vos instances de plan de contrôle Kubernetes. Avec un groupe de placement par répartition utilisant la répartition au niveau des hôtes, toutes les instances de plan de contrôle Kubernetes s'exécutent sur des hôtes distincts au sein des racks sous-jacents disponibles dans votre déploiement Outpost.
- Déploiements avec au moins trois racks physiques au sein d'un seul Outpost logique : au moins trois hôtes doivent être configurés avec le type d'instance que vous choisissez pour vos instances de plan de contrôle Kubernetes. Avec un groupe de placement par répartition utilisant la répartition au niveau des racks, toutes les instances de plan de contrôle Kubernetes s'exécutent sur des racks distincts dans votre déploiement Outpost. Vous pouvez également utiliser le groupe de placement par répartition au niveau des hôtes comme décrit dans l'option précédente.

La création du groupe de placement souhaité vous incombe. Vous spécifiez le groupe de placement lorsque vous appelez l'API `CreateCluster`. Pour plus d'informations sur les groupes de placement et sur la manière de les créer, consultez la section Groupes de placement dans le guide de l'utilisateur Amazon EC2.

Considérations

- Lorsqu'un groupe de placement est spécifié, de la capacité doit être disponible sur votre Outpost pour créer un cluster Amazon EKS local. La capacité varie selon que vous utilisez le type de répartition au niveau des hôtes ou des racks. Si la capacité est insuffisante, le cluster reste à l'état `Creating`. Vous pouvez voir le message `Insufficient Capacity Error` dans le champ de santé de la réponse de l'API `DescribeCluster`. Vous devez libérer de la capacité pour que le processus de création puisse progresser.
- Lors des mises à jour de la plateforme et de la version du cluster local Amazon EKS, les instances de plan de contrôle Kubernetes de votre cluster sont remplacées par de nouvelles instances selon une stratégie de mise à jour continue. Au cours de ce processus de remplacement, chaque instance de plan de contrôle est arrêtée et son emplacement est libéré. Une nouvelle instance mise à jour est provisionnée à sa place. L'instance mise à jour peut être placée dans l'emplacement qui

a été libéré. Si l'emplacement est occupé par une autre instance indépendante et que la capacité disponible est insuffisante pour répondre aux exigences topologiques de répartition, le cluster reste à l'état `Updating`. Vous pouvez voir le message `Insufficient Capacity Error` correspondant dans le champ de santé de la réponse de l'API `DescribeCluster`. Vous devez libérer de la capacité pour que le processus de mise à jour puisse progresser et rétablir les niveaux de haute disponibilité antérieurs.

- Vous pouvez créer un maximum de 500 groupes de placement par compte dans chacun d'eux Région AWS. Pour plus d'informations, consultez la section Règles générales et limitations du guide de l'utilisateur Amazon EC2.

Résolution des problèmes de clusters locaux pour Amazon EKS sur AWS Outposts

Cette rubrique traite de certaines erreurs courantes que vous pourriez rencontrer lorsque vous utilisez des clusters locaux, ainsi que des solutions. Les clusters locaux sont similaires aux clusters Amazon EKS dans le cloud, mais il existe quelques différences dans la façon dont ils sont gérés par Amazon EKS.

Comportement de l'API

Les clusters locaux sont créés via l'API Amazon EKS, mais sont exécutés de manière asynchrone. Cela signifie que les demandes adressées à l'API Amazon EKS aboutissent immédiatement pour les clusters locaux. Toutefois, ces demandes peuvent aboutir, effectuer une interruption immédiate en raison d'erreurs de validation d'entrée, ou échouer et comporter des erreurs de validation descriptives. Ce comportement est similaire à celui de l'API Kubernetes.

Les clusters locaux ne passent pas à l'état `FAILED`. Amazon EKS tente de rapprocher l'état du cluster de l'état souhaité demandé par l'utilisateur de manière continue. Par conséquent, un cluster local peut rester dans l'état `CREATING` pendant une longue période, jusqu'à ce que le problème sous-jacent soit résolu.

Décrire le champ de santé du cluster

Les problèmes liés aux clusters locaux peuvent être découverts à l'aide de la commande `describe-cluster` de l'AWS CLI Amazon EKS. Les problèmes liés aux clusters locaux sont signalés par le champ `cluster.health` de la réponse de la commande `describe-cluster`. Le message contenu dans ce champ inclut un code d'erreur, un message descriptif et les ID de ressources

connexes. Ces informations sont disponibles via l'API et la AWS CLI Amazon EKS uniquement. Dans la commande suivante, remplacez *my-cluster* par le nom de votre cluster local.

```
aws eks describe-cluster --name my-cluster --query 'cluster.health'
```

L'exemple qui suit illustre un résultat.

```
{
    "issues": [
        {
            "code": "ConfigurationConflict",
            "message": "The instance type 'm5.large' is not supported in Outpost 'my-outpost-arn'.",
            "resourceIds": [
                "my-cluster-arn"
            ]
        }
    ]
}
```

Si le problème est irréparable, vous devrez peut-être supprimer le cluster local et en créer un nouveau. Par exemple, essayez d'allouer un cluster avec un type d'instance qui n'est pas disponible sur votre Outpost. Le tableau suivant répertorie les erreurs courantes liées à l'état.

Scénario d'erreur	Code	Message	ResourceIds
Les sous-réseaux fournis sont introuvables.	`ResourceNotFound`	`The subnet ID subnet-id does not exist`	Tous les ID de sous-réseaux fournis
Les sous-réseaux fournis n'appartiennent pas au même VPC.	`ConfigurationConflict`	`Subnets specified must belong to the same VPC`	Tous les ID de sous-réseaux fournis
Certains sous-réseaux fournis n'appartiennent pas à l'Outpost spécifié.	`ConfigurationConflict`	`Subnet subnet-id expected to be in outpost-arn ,`	ID de sous-réseau problématique

Scénario d'erreur	Code	Message	ResourceIds
		but is in *other-outpost-arn*	
Certains sous-réseaux fournis n'appartiennent à aucun Outpost.	ConfigurationConflict	Subnet *subnet-id* is not part of any Outpost	ID de sous-réseau problématique
Certains sous-réseaux fournis ne disposent pas de suffisamment d'adresses libres pour créer des interfaces réseau Elastic pour les instances du plan de contrôle.	ResourceLimitExceeded	The specified subnet does not have enough free addresses to satisfy the request.	ID de sous-réseau problématique
Le type d'instance du plan de contrôle spécifié n'est pas pris en charge sur votre Outpost.	ConfigurationConflict	The instance type *type* is not supported in Outpost *outpost-arn*	ARN de cluster

Scénario d'erreur	Code	Message	ResourceIds
Vous avez résilié une instance Amazon EC2 du plan de contrôle ou la commande `run-instance` a réussi, mais l'état observé est passé à `Terminated`. Cela peut se produire pendant un certain temps une fois que votre Outpost se reconnecte et que des erreurs internes d'Amazon EBS entraînent l'échec d'un flux interne Amazon EC2.	`InternalFailure`	`EC2 instance state "Terminated" is unexpected`	ARN de cluster
Vous avez une capacité insuffisante sur votre Outpost. Cela peut également se produire lors de la création d'un cluster si un Outpost est déconnecté de l'Région AWS.	`ResourceLimitExceeded`	`There is not enough capacity on the Outpost to launch or start the instance.`	ARN de cluster
Votre compte a dépassé votre quota de groupes de sécurité.	`ResourceLimitExceeded`	Message d'erreur renvoyé par l'API Amazon EC2	ID du VPC cible

Scénario d'erreur	Code	Message	ResourceIds
Votre compte a dépassé votre quota d'interface réseau Elastic.	`ResourceLimitExceeded`	Message d'erreur renvoyé par l'API Amazon EC2	l'ID du sous-réseau cible
Les instances du plan de contrôle n'étaient pas accessibles via AWS Systems Manager. Pour la résolution, consultez Les instances du plan de contrôle ne sont pas accessibles via AWS Systems Manager..	`ClusterUnreachable`	Les instances du plan de contrôle Amazon EKS ne sont pas accessibles via SSM. Vérifiez votre SSM et votre configuration réseau, et consultez la documentation de résolution des problèmes EKS sur les Outposts.	ID de l'instance Amazon EC2
Une erreur s'est produite lors de l'obtention de détails pour un groupe de sécurité géré ou une interface réseau Elastic.	Basée sur le code d'erreur du client Amazon EC2.	Message d'erreur renvoyé par l'API Amazon EC2	Tous les ID des groupes de sécurité gérés
Une erreur s'est produite lors de l'autorisation ou de la révocation des règles d'entrée du groupe de sécurité. Cela s'applique à la fois aux groupes de sécurité du cluster et du plan de contrôle.	Basée sur le code d'erreur du client Amazon EC2.	Message d'erreur renvoyé par l'API Amazon EC2	ID de groupe de sécurité problématique

Scénario d'erreur	Code	Message	ResourceIds
Une erreur s'est produite lors de la suppression d'une interface réseau Elastic d'une instance du plan de contrôle.	Basée sur le code d'erreur du client Amazon EC2.	Message d'erreur renvoyé par l'API Amazon EC2	ID d'interface réseau Elastoc problématique

Le tableau suivant répertorie les erreurs provenant d'autres Services AWS qui sont présentes dans le champ de santé de la réponse `describe-cluster`.

Code d'erreur Amazon EC2	Code de problème de santé du cluster	Description
`AuthFailure`	`AccessDenied`	Cette erreur peut se produire pour différentes raisons. La raison la plus courante est que vous avez accidentellement supprimé une balise que le service utilise pour réduire la portée de la politique de rôle lié à un service du plan de contrôle. Si cela se produit, Amazon EKS ne peut plus gérer et surveiller ces ressources AWS.
`UnauthorizedOperation`	`AccessDenied`	Cette erreur peut se produire pour différentes raisons. La raison la plus courante est que vous avez accidentellement supprimé une balise que le service utilise pour réduire la portée de la politique de rôle lié à un service du

Code d'erreur Amazon EC2	Code de problème de santé du cluster	Description
		plan de contrôle. Si cela se produit, Amazon EKS ne peut plus gérer et surveiller ces ressources AWS.
`InvalidSubnetID.NotFound`	`ResourceNotFound`	Cette erreur se produit lorsque l'ID de sous-réseau pour les règles d'entrée d'un groupe de sécurité est introuvable.
`InvalidPermission.NotFound`	`ResourceNotFound`	Cette erreur se produit lorsque les autorisations pour les règles d'entrée d'un groupe de sécurité ne sont pas correctes.
`InvalidGroup.NotFound`	`ResourceNotFound`	Cette erreur se produit lorsque le groupe des règles d'entrée d'un groupe de sécurité est introuvable.
`InvalidNetworkInterfaceID.NotFound`	`ResourceNotFound`	Cette erreur se produit lorsque l'ID de l'interface réseau pour les règles d'entrée d'un groupe de sécurité est introuvable.
`InsufficientFreeAddressesInSubnet`	`ResourceLimitExceeded`	Cette erreur se produit lorsque le quota de ressources du sous-réseau est dépassé.
`InsufficientCapacityOnOutpost`	`ResourceLimitExceeded`	Cette erreur se produit lorsque le quota de capacité de l'avant-poste est dépassé.
`NetworkInterfaceLimitExceeded`	`ResourceLimitExceeded`	Cette erreur se produit lorsque le quota de l'interface réseau Elastic est dépassé.

Code d'erreur Amazon EC2	Code de problème de santé du cluster	Description
`SecurityGroupLimitExceeded`	`ResourceLimitExceeded`	Cette erreur se produit lorsque le quota du groupe de sécurité est dépassé.
`VcpuLimitExceeded`	`ResourceLimitExceeded`	Cette erreur est observée lors de la création d'une instance Amazon EC2 dans un nouveau compte. L'erreur peut être similaire à ce qui suit : "`You have requested more vCPU capacity than your current vCPU limit of 32 allows for the instance bucket that the specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit.`"
`InvalidParameterValue`	`ConfigurationConflict`	Amazon EC2 renvoie ce code d'erreur si le type d'instance spécifié n'est pas pris en charge sur l'Outpost.
Toutes les autres erreurs	`InternalFailure`	Aucune

Impossible de créer ou de modifier des clusters

Les clusters locaux nécessitent des autorisations et des politiques différentes de celles des clusters Amazon EKS qui sont hébergés dans le cloud. Lorsqu'un cluster ne parvient pas à se créer et génère

une `InvalidPermissions` erreur, vérifiez que le rôle de cluster que vous utilisez est associé à la politique gérée LocalOutpostClusterPolicy Amazoneks. Tous les autres appels d'API nécessitent le même ensemble d'autorisations que les clusters Amazon EKS dans le cloud.

Le cluster est bloqué à l'état **`CREATING`**

Le temps nécessaire à la création d'un cluster local varie en fonction de plusieurs facteurs. Ces facteurs comprennent la configuration de votre réseau, la configuration d'Outpost et la configuration du cluster. En général, un cluster local est créé et passe à l'état `ACTIVE` dans les 15 à 20 minutes. Si un cluster local reste dans l'état `CREATING`, vous pouvez appeler `describe-cluster` pour obtenir des informations sur la cause dans le champ de sortie `cluster.health`.

Les problèmes les plus courants sont les suivants :

AWS Systems Manager (Systems Manager) rencontre les problèmes suivants :

- Votre cluster ne peut pas se connecter à l'instance du plan de contrôle à partir de l'Région AWS dans laquelle se trouve Systems Manager. Vous pouvez le vérifier en appelant `aws ssm start-session --target` *`instance-id`* depuis un hôte bastion de la région. Si cette commande ne fonctionne pas, vérifiez si Systems Manager est exécuté sur l'instance du plan de contrôle. Une autre solution consiste à supprimer le cluster, puis à le recréer.
- Il se peut que les instances du plan de contrôle de Systems Manager n'aient pas accès à Internet. Vérifiez si le sous-réseau que vous avez fourni lors de la création du cluster possède une passerelle NAT et un VPC avec une passerelle Internet. Utilisez l'analyseur d'accessibilité VPC pour vérifier si l'instance du plan de contrôle peut atteindre la passerelle Internet. Pour plus d'informations, consultez Getting started with VPC Reachability Analyzer (langue française non garantie).
- Le rôle ARN que vous avez fourni ne contient pas de politiques. Vérifiez si la politique AWS politique gérée : Amazoneks LocalOutpostClusterPolicy a été retirée du rôle. Cela peut également se produire si une pile AWS CloudFormation est mal configurée.

Plusieurs sous-réseaux mal configurés et spécifiés lors de la création d'un cluster :

- Tous les sous-réseaux fournis doivent être associés au même Outpost et pouvoir communiquer entre eux. Lorsque plusieurs sous-réseaux sont spécifiés lors de la création d'un cluster, Amazon EKS tente de répartir les instances du plan de contrôle sur plusieurs sous-réseaux.

- Les groupes de sécurité gérés par Amazon EKS sont appliqués sur l'interface réseau Elastic. Cependant, d'autres éléments de configuration tels que les règles de pare-feu NACL peuvent entrer en conflit avec les règles de l'interface réseau Elastic.

La configuration du VPC et du DNS du sous-réseau est mal configurée ou manquante

Consultez Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux.

Impossible de joindre des nœuds à un cluster

Causes courantes :

- Problèmes AMI :
  - Vous utilisez une AMI non prise en charge. Vous devez utiliser v20220620 ou une version ultérieure pour une AMI Amazon Linux optimisées pour Amazon EKS Amazon Linux optimisée par Amazon EKS.
  - Si vous avez utilisé un modèle AWS CloudFormation pour créer vos nœuds, assurez-vous qu'il n'utilisait pas une AMI non prise en charge.
- L'authentificateur IAM `ConfigMap` AWS manquant – Vous devez le créer le cas échéant. Pour plus d'informations, consultez Appliquer la `ConfigMap aws-auth` à votre cluster.
- Le mauvais groupe de sécurité est utilisé – Assurez-vous d'utiliser `eks-cluster-sg-`*`cluster-name`*`-`*`uniqueid`* pour le groupe de sécurité de vos composants master. Le groupe de sécurité sélectionné est modifié par AWS CloudFormation pour autoriser la création d'un nouveau groupe de sécurité chaque fois que la pile est utilisée.
- Suite à des étapes inattendues liées à un VPC de lien privé – Des données CA incorrectes (`--b64-cluster-ca`) ou le mauvais point de terminaison de l'API (`--apiserver-endpoint`) sont transmis.
- Politique de sécurité de Pod mal configurée :
  - Les ensembles de démons CoreDNS et Amazon VPC CNI plugin for Kubernetes doivent s'exécuter sur les nœuds pour que les nœuds puissent rejoindre le cluster et communiquer avec lui.
  - Le Amazon VPC CNI plugin for Kubernetes nécessite certaines fonctions réseau privilégiées pour fonctionner correctement. Vous pouvez afficher les fonctions réseau privilégiées à l'aide de la commande suivante : `kubectl describe psp eks.privileged`.

Nous ne recommandons pas de modifier la politique de sécurité du pod par défaut. Pour plus d'informations, consultez Politique de sécurité de pod.

Collecte des journaux

Lorsqu'un Outpost se déconnecte de l'Région AWS à laquelle il est associé, le cluster Kubernetes continuera probablement à fonctionner normalement. Toutefois, si le cluster ne fonctionne pas correctement, suivez les étapes de dépannage dans Préparation aux déconnexions du réseau. Si vous rencontrez d'autres problèmes, contactez AWS Support. AWS Support peut vous guider pour télécharger et exécuter un outil de collecte de journaux. De cette façon, vous pouvez collecter les journaux de vos instances du plan de contrôle du cluster Kubernetes et les envoyer au support AWS Support pour un examen plus approfondi.

Les instances du plan de contrôle ne sont pas accessibles via AWS Systems Manager.

Lorsque les instances du plan de contrôle Amazon EKS ne sont pas accessibles via AWS Systems Manager (Systems Manager), Amazon EKS affiche l'erreur suivante pour votre cluster.

```
Amazon EKS control plane instances are not reachable through SSM. Please verify your
 SSM and network configuration, and reference the EKS on Outposts troubleshooting
 documentation.
```

Pour résoudre ce problème, assurez-vous que votre VPC et vos sous-réseaux répondent aux exigences figurant dans Exigences et considérations Amazon EKS requises pour le VPC et les sous-réseaux des clusters locaux et que vous avez suivi les étapes de la section Configuration de Session Manager du Guide de l'utilisateur AWS Systems Manager.

Lancement de nœuds Amazon Linux autogérés sur un Outpost

Cette rubrique décrit comment lancer des groupes Auto Scaling de nœuds Amazon Linux sur un Outpost qui s'enregistrent auprès de votre cluster Amazon EKS. Le cluster peut se trouver sur AWS Cloud ou sur un avant-poste.

Prérequis

- Un Outpost existant. Pour plus d'informations, consultez What is AWS Outposts.

- Un cluster Amazon EKS existant. Pour déployer un cluster sur le AWS Cloud, voirCréation d'un cluster Amazon EKS. Pour déployer un cluster sur un Outpost, consultez Clusters locaux pour Amazon EKS sur AWS Outposts.
- Supposons que vous créez vos nœuds dans un cluster sur le AWS Cloud et que vous ayez des sous-réseaux Région AWS là où vous avez activé AWS Outposts les AWS Wavelength Zones AWS Locales. Alors, ces sous-réseaux ne doivent pas avoir été transmis lors de la création de votre cluster. Si vous créez vos nœuds dans un cluster sur un Outpost, vous devez avoir transmis un sous-réseau Outpost lors de la création de votre cluster.
- (Recommandé pour les clusters situés sur le AWS Cloud) Le Amazon VPC CNI plugin for Kubernetes module complémentaire est configuré avec son propre rôle IAM auquel est attachée la politique IAM nécessaire. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA). Les clusters locaux ne prennent pas en charge les rôles IAM pour les comptes de service.

Vous pouvez créer un groupe de nœuds Amazon Linux autogéré avec `eksctl` ou AWS Management Console (avec un AWS CloudFormation modèle). Vous pouvez également utiliser Terraform.

eksctl

Prérequis

Version `0.183.0` ou ultérieure de l'outil de ligne de commande `eksctl` installée sur votre appareil ou AWS CloudShell. Pour installer ou mettre à jour `eksctl`, veuillez consulter Installation dans la documentation de `eksctl`.

Pour lancer des nœuds Linux autogérés à l'aide de **`eksctl`**

1. Si votre cluster est sur le AWS Cloud et que la politique IAM gérée AmazonEKS_CNI_Policy est attachée à votre Rôle IAM de nœud Amazon EKS, nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service `aws-node` Kubernetes à la place. Pour plus d'informations, consultez Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA). Si votre cluster se trouve sur votre Outpost, la politique doit être associée à votre rôle de nœud.
2. La commande suivante crée un groupe de nœuds dans un cluster existant. Le cluster doit avoir été créé à l'aide d'`eksctl`. Remplacer *`al-nodes`* avec un nom pour votre groupe de nœuds. Le nom du groupe de nœuds ne peut pas dépasser 63 caractères. Il doit

commencer par une lettre ou un chiffre, mais peut également inclure des tirets et des traits de soulignement pour les autres caractères. Remplacez *my-cluster* par le nom de votre cluster. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster. Si votre cluster existe sur un Outpost, remplacez *id* par l'ID d'un sous-réseau Outpost. Si votre cluster existe sur le AWS Cloud, remplacez-le *id* par l'ID d'un sous-réseau que vous n'avez pas spécifié lors de la création de votre cluster. Remplacez *instance-type* par un type d'instance pris en charge par votre Outpost. Remplacez les valeurs de *example values* restantes par vos propres valeurs. Par défaut, les nœuds sont créés avec la même version de Kubernetes que le plan de contrôle.

Remplacez *instance-type* par un type d'instance disponible sur votre Outpost.

Remplacez *my-key* par le nom de votre paire de clés Amazon EC2 ou de votre clé publique. Cette clé est utilisée pour SSH dans vos nœuds après leur lancement. Si vous ne possédez pas d'une paire de clés Amazon EC2, vous pouvez en créer une dans la AWS Management Console. Pour plus d'informations, veuillez consulter la rubrique Paires de clés Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Créez votre groupe de nœuds avec la commande suivante.

```
eksctl create nodegroup --cluster my-cluster --name al-nodes --node-
type instance-type \
    --nodes 3 --nodes-min 1 --nodes-max 4 --managed=false --node-volume-type gp2
 --subnet-ids subnet-id
```

Si votre cluster est déployé sur AWS Cloud :

- Le groupe de nœuds que vous déployez peut attribuer des adresses `IPv4` aux Pods à partir d'un bloc CIDR différent de celui de l'instance. Pour plus d'informations, consultez Mise en réseau personnalisée pour les pods.
- Le groupe de nœuds que vous déployez ne nécessite pas d'accès Internet sortant. Pour plus d'informations, consultez Exigences relatives aux clusters privés.

Pour voir la liste complète des options disponibles et des valeurs par défaut, consultez AWS Outposts Support dans la documentation `eksctl`.

Si les nœuds ne parviennent pas à rejoindre le cluster, consultez Les nœuds ne parviennent pas à joindre le cluster dans Dépannage d'Amazon EKS et Impossible de joindre des nœuds à un cluster dans Résolution des problèmes de clusters locaux pour Amazon EKS sur AWS Outposts.

L'exemple qui suit illustre un résultat. Plusieurs lignes sont affichées pendant la création des nœuds. L'une des dernières lignes de sortie est similaire à la ligne d'exemple suivante.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facultatif) Déployez un exemple d'application pour tester votre cluster et les nœuds Linux.

AWS Management Console

Étape 1 : Pour lancer des nœuds Amazon Linux autogérés à l'aide du AWS Management Console

1. Téléchargez la dernière version du AWS CloudFormation modèle.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/
amazon-eks-nodegroup.yaml
```

2. Ouvrez la AWS CloudFormation console à l'adresse https://console.aws.amazon.com/cloudformation.
3. Choisissez Create stack (Créer une pile), puis sélectionnez Avec de nouvelles ressources (standard).
4. Pour Spécifier un modèle, sélectionnez Upload a template file (Télécharger un fichier de modèle), puis sélectionnez Choose file (Choisir un fichier). Sélectionnez le fichier `amazon-eks-nodegroup.yaml` que vous avez téléchargé à une étape précédente, puis sélectionnez Next (Suivant).
5. Dans la page Specify stack details (Spécifier les détails de la pile), saisissez les paramètres suivants, puis choisissez Next (Suivant) :

   - Nom de la pile : choisissez un nom pour votre pile AWS CloudFormation . Par exemple, vous pouvez l'appeler ***al-nodes***. Un nom ne peut contenir que des caractères alphanumériques (sensibles à la casse) et des traits d'union. Il doit commencer par un caractère alphanumérique et ne doit pas dépasser 100 caractères. Le nom doit être unique dans le Région AWS et dans Compte AWS lequel vous créez le cluster.

- ClusterName: Entrez le nom de votre cluster. Si ce nom ne correspond pas à celui de votre cluster, vos nœuds ne peuvent pas rejoindre le cluster.
- ClusterControlPlaneSecurityGroupe : Choisissez la SecurityGroupsvaleur à partir de la AWS CloudFormation sortie que vous avez générée lors de la création de votre VPC.

  Les étapes suivantes montrent une opération permettant de récupérer le groupe applicable.

  1. Ouvrez la console Amazon EKS à l'adresse https://console.aws.amazon.com/eks/home#/clusters.
  2. Choisissez le nom du cluster.
  3. Choisissez l'onglet Networking (Mise en réseau).
  4. Utilisez la valeur Groupes de sécurité supplémentaires comme référence lorsque vous effectuez une sélection dans la liste déroulante des ClusterControlPlaneSecuritygroupes.
- NodeGroupNom : entrez le nom de votre groupe de nœuds. Ce nom peut être utilisé ultérieurement pour identifier le groupe de nœuds Auto Scaling qui est créé pour vos nœuds.
- NodeAutoScalingGroupMinSize: Entrez le nombre minimum de nœuds que votre groupe Auto Scaling de nœuds peut atteindre.
- NodeAutoScalingGroupDesiredCapacity: Entrez le nombre de nœuds que vous souhaitez atteindre lors de la création de votre pile.
- NodeAutoScalingGroupMaxSize: Entrez le nombre maximum de nœuds que votre groupe Auto Scaling de nœuds peut atteindre.
- NodeInstanceType : Choisissez un type d'instance pour vos nœuds. Si votre cluster s'exécute sur le AWS Cloud, pour plus d'informations, consultezChoix d'un type d'instance Amazon EC2. Si votre cluster s'exécute sur un Outpost, vous ne pouvez sélectionner qu'un type d'instance disponible sur votre Outpost.
- NodeImageIDSSMParam : prérempli avec le paramètre Amazon EC2 Systems Manager d'une AMI récemment optimisée pour Amazon EKS pour une version variable. Kubernetes Pour utiliser une autre version mineure de Kubernetes prise en charge avec Amazon EKS, remplacez *1.XX* par une autre version prise en charge. Nous vous recommandons de spécifier la même version de Kubernetes que celle de votre cluster.

Pour utiliser l'AMI accélérée optimisée pour Amazon EKS, remplacez *amazon-linux-2* par **amazon-linux-2-gpu**. Pour utiliser l'AMI Arm optimisée pour Amazon EKS, remplacez *amazon-linux-2* par **amazon-linux-2-arm64**.

> **Note**
>
> L'AMI du nœud Amazon EKS est basée sur Amazon Linux. Vous pouvez suivre les évènements de sécurité et de confidentialité pour Amazon Linux 2 via le centre de sécurité Amazon Linux ou souscrire au flux RSS associé. Les événements de sécurité et de confidentialité incluent une présentation du problème, les packages concernés et la manière de mettre à jour vos instances pour résoudre le problème.

- NodeImageID : (Facultatif) Si vous utilisez votre propre AMI personnalisée (au lieu de l'AMI optimisée pour Amazon EKS), entrez un ID d'AMI de nœud pour votre Région AWS. Si vous spécifiez une valeur ici, elle remplace toutes les valeurs du champ NodeImageIDSSMParam.
- NodeVolumeTaille : Spécifiez une taille de volume racine pour vos nœuds, en GiB.
- NodeVolumeType : Spécifiez un type de volume racine pour vos nœuds.
- KeyName: Entrez le nom d'une paire de clés SSH Amazon EC2 que vous pourrez utiliser pour vous connecter via SSH à vos nœuds après leur lancement. Si vous ne possédez pas déjà une paire de clés Amazon EC2, vous pouvez en créer une dans l' AWS Management Console. Pour plus d'informations, veuillez consulter la rubrique Paires de clés Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

  > **Note**
  >
  > Si vous ne fournissez pas de paire de clés ici, la création de la AWS CloudFormation pile échoue.

- BootstrapArguments: Il existe plusieurs arguments facultatifs que vous pouvez transmettre à vos nœuds. Pour de plus amples informations, veuillez consulter Utilisation du script d'amorçage sur GitHub. Si vous ajoutez des nœuds à un cluster local Amazon EKS AWS Outposts (sur lequel s'exécutent les instances du plan de Kubernetes contrôle AWS Outposts) et que le cluster ne dispose pas de connexion Internet d'entrée et de sortie (également appelée clusters privés), vous devez fournir les arguments bootstrap suivants (sur une seule ligne).

```
--b64-cluster-ca ${CLUSTER_CA} --apiserver-endpoint https://
${APISERVER_ENDPOINT} --enable-local-outpost true --cluster-id ${CLUSTER_ID}
```

- DisableIMDSv1 : par défaut, chaque nœud prend en charge le service de métadonnées d'instance version 1 (IMDSv1) et IMDSv2. Vous pouvez désactiver IMDSv1. Pour empêcher les futurs nœuds et les Pods du groupe de nœuds d'utiliser IMDSv1, définissez DisableIMDSv1 sur true. Pour de plus amples informations au sujet d'IMDS, consultez Configuration du service des métadonnées d'instance. Pour plus d'informations sur la façon d'en restreindre l'accès sur vos nœuds, consultez Restreindre l'accès au profil d'instance affecté au composant master.
- VpcId: Entrez l'ID du VPC que vous avez créé. Avant de choisir un VPC, consultez Exigences et considérations requises pour le VPC.
- Subnets : si votre cluster se trouve sur un Outpost, choisissez au moins un sous-réseau privé dans votre VPC. Avant de choisir les sous-réseaux, consultez Exigences et considérations requises pour les sous-réseaux. Vous pouvez consulter les sous-réseaux privés en ouvrant le lien de chaque sous-réseau depuis l'onglet Networking (Mise en réseau) de votre cluster.

6. Sélectionnez les choix que vous souhaitez sur la page Configure stack options (Configurer les options de la pile), puis choisissez Next (Suivant).
7. Cochez la case à gauche de Je comprends que AWS CloudFormation pourrait créer des ressources IAM., puis choisissez Create stack (Créer une pile).
8. Lorsque la création de votre pile est terminée, sélectionnez la pile dans la console et choisissez Outputs (Sorties).
9. Enregistrez le NodeInstancerôle du groupe de nœuds créé. Vous en aurez besoin lors de la configuration de vos nœuds pour Amazon EKS.

Étape 2 : pour autoriser les nœuds à rejoindre votre cluster

1. Vérifiez si vous avez déjà appliqué le `ConfigMap` `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si vous voyez un `ConfigMap` `aws-auth`, mettez-le à jour si nécessaire.

   a. Ouvrez le `ConfigMap` pour le modifier.

```
kubectl edit -n kube-system configmap/aws-auth
```

b. Ajoutez une nouvelle entrée `mapRoles` si nécessaire. Définissez la `rolearn` valeur sur la valeur du NodeInstancerôle que vous avez enregistrée lors de la procédure précédente.

```
[...]
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

c. Enregistrez le fichier et quittez votre éditeur de texte.

3. Si vous avez reçu un message d'erreur indiquant « `Error from server (NotFound): configmaps "aws-auth" not found` », appliquez le stock `ConfigMap`.

a. Téléchargez la mappe de configuration.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm.yaml
```

b. Dans le `aws-auth-cm.yaml` fichier, définissez la `rolearn` valeur du NodeInstancerôle que vous avez enregistrée lors de la procédure précédente. Pour ce faire, utilisez un éditeur de texte ou remplacez *`my-node-instance-role`* et exécutez la commande suivante :

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-
instance-role|' aws-auth-cm.yaml
```

c. Appliquez la configuration. L'exécution de cette commande peut prendre quelques minutes.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Observez le statut de vos nœuds et attendez qu'ils obtiennent le statut `Ready`.

```
kubectl get nodes --watch
```

Saisissez `Ctrl+C` pour revenir à une invite de shell.

> **Note**
>
> Si vous recevez d'autres erreurs concernant les types d'autorisations ou de ressources, consultez Accès non autorisé ou refusé (`kubectl`) dans la rubrique relative à la résolution des problèmes.

Si les nœuds ne parviennent pas à rejoindre le cluster, consultez Les nœuds ne parviennent pas à joindre le cluster dans Dépannage d'Amazon EKS et Impossible de joindre des nœuds à un cluster dans Résolution des problèmes de clusters locaux pour Amazon EKS sur AWS Outposts.

5. Installez le pilote CSI Amazon EBS. Pour plus d'informations, consultez la section Installation sur GitHub. Dans la section Set up driver permission (Configurer les autorisations du pilote), assurez-vous de suivre les instructions de l'option Using IAM instance profile (Utilisation du profil d'instance IAM). Vous devez utiliser la classe de stockage `gp2`. La classe de stockage `gp3` n'est pas prise en charge.

   Pour créer une classe de stockage `gp2` sur votre cluster, suivez les étapes suivantes.

   1. Exécutez la commande ci-dessous pour créer un fichier`gp2-storage-class.yaml`.

```
cat >gp2-storage-class.yaml <<EOF
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
  name: ebs-sc
provisioner: ebs.csi.aws.com
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp2
  encrypted: "true"
allowVolumeExpansion: true
```

```
EOF
```

2. Appliquez le manifeste à votre cluster.

```
kubectl apply -f gp2-storage-class.yaml
```

6. (Nœuds GPU uniquement) Si vous avez opté pour une instance de type GPU et l'AMI accélérée optimisée pour Amazon EKS, vous devez mettre en œuvre le plugin de périphérique NVIDIA pour Kubernetes en tant que DaemonSet sur votre cluster. Remplacez *vX.X.X* par la version NVIDIA/k8s-device-plugin souhaitée avant d'exécuter la commande suivante.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-
plugin/vX.X.X/nvidia-device-plugin.yml
```

Étape 3 : actions supplémentaires

1. (Facultatif) Déployez un exemple d'application pour tester votre cluster et les nœuds Linux.
2. Si votre cluster est déployé sur un Outpost, ignorez cette étape. Si votre cluster est déployé sur le AWS Cloud, les informations suivantes sont facultatives. Si la politique d'IAM gérée AmazonEKS_CNI_Policy est associée à votre Rôle IAM de nœud Amazon EKS, nous vous recommandons de l'attribuer à un rôle IAM que vous associez au compte de service `aws-node` Kubernetes à la place. Pour plus d'informations, voir Configuration de l'utilisation Amazon VPC CNI plugin for Kubernetes des rôles IAM pour les comptes de service (IRSA).

Projets connexes

Ces projets open source étendent les fonctionnalités des clusters Kubernetes s'exécutant sur et en dehors de AWS, y compris les clusters gérés par Amazon EKS.

Outils de gestion

Outils de gestion connexes pour les clusters Amazon EKS et Kubernetes.

eksctl

`eksctl` est un outil de CLI pour créer des clusters sur Amazon EKS.

- URL du projet
- Documentation du projet
- Blog open source AWS : eksctl: Amazon EKS Cluster with One Command

Contrôleurs AWS pour Kubernetes

Avec les contrôleurs AWS pour Kubernetes, vous pouvez créer et gérer des ressources AWS directement à partir de votre cluster Kubernetes.

- URL du projet
- Blog open source AWS : opérateur de service AWS pour Kubernetes maintenant disponible

Flux CD

Flux est un outil que vous pouvez utiliser pour gérer la configuration de votre cluster à l'aide de Git. Il utilise un opérateur dans le cluster pour déclencher des déploiements à l'intérieur de Kubernetes. Pour plus d'informations sur les opérateurs, consultez OperatorHub.io (français non disponible) sur GitHub.

- URL du projet
- Documentation du projet

CDK pour Kubernetes

Avec le CDK pour Kubernetes (cdk8s), vous pouvez définir des applications et des composants Kubernetes en utilisant des langages de programmation familiers. Les applications cdk8s se synthétisent en manifestes Kubernetes standards, qui peuvent être appliqués à n'importe quel cluster Kubernetes.

- URL du projet
- Documentation du projet
- Blog des conteneurs AWS : Présentation de cdk8s+ : API pilotées par intention pour les objets Kubernetes

Réseaux

Projets de mise en réseau connexes pour les clusters Amazon EKS et Kubernetes.

Amazon VPC CNI plugin for Kubernetes

Amazon EKS prend en charge la mise en réseau VPC native via Amazon VPC CNI plugin for Kubernetes. Le plugin attribue une adresse IP depuis votre VPC à chaque Pod.

- URL du projet
- Documentation du projet

AWS Load Balancer Controller pour Kubernetes

Le AWS Load Balancer Controller permet de gérer les AWS Elastic Load Balancers pour un cluster Kubernetes. Il satisfait les ressources d'entrée Kubernetes en allouant les AWS Application Load Balancers. Il satisfait les ressources du service Kubernetes en approvisionnant les AWS Network Load Balancers.

- URL du projet
- Documentation du projet

ExternalDNS

ExternalDNS synchronise les services et les entrées Kubernetes exposés avec des fournisseurs DNS, notamment Amazon Route 53 et AWS Service Discovery.

- URL du projet
- Documentation du projet

Machine learning

Projets de machine learning connexes pour les clusters Amazon EKS et Kubernetes.

Kubeflow

Boîte à outils de machine learning pour Kubernetes.

- URL du projet
- Documentation du projet
- Blog open source AWS : Kubeflow on Amazon EKS

Auto Scaling

Projets de scalabilité automatique connexes pour les clusters Amazon EKS et Kubernetes.

Cluster Autoscaler

Cluster Autoscaler est un outil qui ajuste automatiquement la taille du cluster Kubernetes en fonction de la sollicitation du processeur et de la mémoire.

- URL du projet
- Documentation du projet
- Atelier Amazon EKS : https://www.eksworkshop.com/

Escalator

Escalator est un outil de scalabilité automatique horizontal optimisé de lot ou de tâche pour Kubernetes.

- URL du projet
- Documentation du projet

Surveillance

Projets de surveillance connexes pour les clusters Amazon EKS et Kubernetes.

Prometheus

Prometheus est une boîte à outils de surveillance de systèmes et d'alerte open source.

- URL du projet
- Documentation du projet
- Atelier Amazon EKS : https://eksworkshop.com/intermediate/240_monitoring/

Intégration continue/déploiement continu

Projets d'intégration/de déploiement en continu (CI/CD) connexes pour les clusters Amazon EKS et Kubernetes.

Jenkins X

Solution d'intégration/de déploiement en continu (CI/CD) pour les applications cloud modernes sur les clusters Amazon EKS et Kubernetes.

- URL du projet
- Documentation du projet

Nouvelles fonctions et feuille de route Amazon EKS

Pour en savoir plus sur les nouvelles fonctions d'Amazon EKS, faites défiler jusqu'au flux Quoi de neuf sur la page Quoi de neuf avec AWS. Vous pouvez également consulter la feuille de route sur GitHub, qui vous permet de connaître les fonctions et les priorités à venir afin que vous puissiez planifier la façon dont vous souhaitez utiliser Amazon EKS à l'avenir. Vous pouvez nous fournir des commentaires directs sur les priorités de la feuille de route.

Historique du document pour Amazon EKS

Le tableau suivant décrit les principales mises à jour et les nouvelles fonctions pour le Guide de l'utilisateur Amazon EKS. Nous mettons aussi la documentation à jour régulièrement pour prendre en compte les commentaires qui nous sont envoyés.

Modification	Description	Date
Kubernetes version `1.30`	Ajout de la prise en charge de la version `1.30` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	23 mai 2024
Mise à jour de la version de la plateforme Amazon EKS	Il s'agit d'une nouvelle version de plateforme qui comporte des correctifs et des améliorations de sécurité. Cela inclut les nouvelles versions de correctif de Kubernetes`1.29.4`,, `1.28.9` et. `1.27.13`	14 mai 2024
à l'CoreDNSéchelle automatique	CoreDNSautoscaler adaptera dynamiquement le nombre de répliques du CoreDNS déploiement dans un cluster EKS en fonction du nombre de nœuds et de cœurs de processeur. Cette fonctionnalité fonctionne pour CoreDNS `v1.9` la dernière version de plate-forme d'EKS `1.25` et les versions ultérieures.	14 mai 2024
CloudWatch Container Insights pour Windows	Le Amazon CloudWatch Observability Operator module complémentaire	10 avril 2024

	autorise désormais également Container Insights les Windows nœuds de travail dans le cluster.	
Kubernetes	Ajout d'une nouvelle rubrique sur les concepts de Kubernetes.	5 avril 2024
Restructurer l'accès et le contenu IAM	Déplacez les pages existantes liées aux sujets relatifs à l'accès et à l'IAM, telles que la carte de configuration d'authentification, les entrées d'accès, l'ID du pod et l'IRSA dans une nouvelle section. Réviser le contenu de l'aperçu.	2 avril 2024
Support du Bottlerocket système d'exploitation pour le pilote Amazon S3 CSI	Le pilote CSI Mountpoint pour Amazon S3 est désormais compatible avec. Bottlerocket	13 mars 2024
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	4 mars 2024
2023	Amazon Linux 2023 (AL2023) est un nouveau système d'exploitation basé sur Linux conçu pour fournir un environnement sécurisé, stable et performant pour vos applications cloud.	29 février 2024

EKS Pod Identity et IRSA prennent en charge les sidecars dans Kubernetes 1.29	En Kubernetes 1.29, les conteneurs sidecar sont disponibles dans les clusters Amazon EKS. Les conteneurs annexes sont pris en charge avec des rôles IAM pour les comptes de service ou EKS Pod Identity. Pour plus d'informations sur les sidecars, consultez la section Containers sidecar dans la documentation. Kubernetes	26 février 2024
Kubernetes version 1.29	Ajout de la prise en charge de la version 1.29 de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	23 janvier 2024
Version complète : Amazon EKS Extended Support pour les Kubernetes versions	Le support étendu des versions de Kubernetes vous permet de conserver une version spécifique de Kubernetes pendant plus de 14 mois.	16 janvier 2024

Détection de l'état du cluster Amazon EKS dans le AWS cloud	Amazon EKS détecte les problèmes liés à vos clusters Amazon EKS et à l'infrastructure des conditions requises pour le cluster en termes de santé du cluster. Vous pouvez consulter les problèmes liés à vos clusters EKS dans AWS Management Console et dans `health` le cluster dans l'API EKS. Ces problèmes s'ajoutent aux problèmes détectés et affichés par la console. Auparavant, l'état de santé des clusters n'était disponible que pour les clusters locaux sur AWS Outposts.	28 décembre 2023
Région AWS	Amazon EKS est désormais disponible dans la Région AWS Canada Ouest (Calgary) (`ca-west-1`).	20 décembre 2023
Informations sur les clusters	Vous pouvez désormais obtenir des recommandations sur votre cluster sur la base de vérifications récurrentes.	20 décembre 2023

Vous pouvez désormais accorder aux utilisateurs et rôles IAM l'accès à votre cluster à l'aide d'entrées d'accès	Avant l'introduction des entrées d'accès, vous avez accordé aux utilisateurs et rôles IAM l'accès à votre cluster en ajoutant des entrées à la `ConfigMap aws-auth`. Chaque cluster dispose désormais d'un mode d'accès, et vous pouvez passer à l'utilisation des entrées d'accès selon votre calendrier. Après avoir changé de mode, vous pouvez ajouter des utilisate urs en ajoutant des entrées d'accès dans les kits de développement logiciel (SDK) et dans les AWS CLI kits de AWS développement logiciel (SDK). AWS CloudFormation	18 décembre 2023
Mise à jour de la version de plateforme Amazon EKS	Il s'agit d'une nouvelle version de plateforme qui comporte des correctifs et des améliorat ions de sécurité. Cela inclut les nouvelles versions de correctif de Kubernetes `1.28.4`, `1.27.8`, `1.26.11` et `1.25.16`.	12 décembre 2023
Mountpoint pour le pilote CSI Amazon S3	Vous pouvez désormais installer le Mountpoint pour le pilote Amazon S3 CSI sur les clusters Amazon EKS.	27 novembre 2023

Activation des métriques Prometheus lors de la création d'un cluster	Dans le AWS Management Console, vous pouvez désormais activer les Prometheus métriques lors de la création d'un cluster. Vous pouvez également voir les détails du scraper Prometheus dans l'onglet Observabilité.	26 novembre 2023
Identités du pod Amazon EKS	Les identités du pod Amazon EKS associent un rôle IAM à un compte de service Kubernetes. Grâce à cette fonction, vous n'avez plus besoin de fournir des autorisations étendues au rôle IAM du nœud. De cette façon, Pods sur ce nœud, vous pouvez appeler AWS des API. Contrairement aux rôles IAM des comptes de service, les identités du pod EKS sont complètement à l'intérieur d'EKS ; vous n'avez pas besoin d'un fournisseur d'identité OIDC.	26 novembre 2023
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	26 novembre 2023
Contrôleur d'instantané CSI	Vous pouvez désormais installer le contrôleur d'instantané CSI pour l'utiliser avec des pilotes CSI compatibles, tels que le pilote CSI Amazon EBS.	17 novembre 2023

Réécriture des rubriques de l'opérateur ADOT	La section sur la prise en charge du module complémentaire Amazon EKS pour l'opérateur ADOT était redondante avec la documentation AWS Distro for OpenTelemetry. Nous avons migré les informations essentielles restantes vers cette ressource afin de réduire les informations obsolètes et incohérentes.	14 novembre 2023
Prise en charge du module complémentaire EKS CoreDNS pour les métriques Prometheus	Les versions `v1.10.1-eksbuild.5` , `v1.9.3-eksbuild.9` et `v1.8.7-eksbuild.8` du module complémentaire EKS pour CoreDNS exposent le port sur lequel CoreDNS publie des métriques, dans le service `kube-dns`. Cela facilite l'inclusion des métriques CoreDNS dans vos systèmes de surveillance.	10 novembre 2023
Module complémentaire Amazon EKS CloudWatch Observability Operator	Ajout de la page Amazon EKS CloudWatch Observability Operator.	6 novembre 2023
Blocs de capacité pour les instances P5 autogérées aux USA Est (Ohio)	Aux USA Est (Ohio), vous pouvez désormais utiliser des blocs de capacité pour les instances P5 autogérées.	31 octobre 2023

Les clusters prennent en charge des sous-réseaux et des groupes de sécurité	Vous pouvez mettre à jour le cluster pour modifier les sous-réseaux et les groupes de sécurité qu'il utilise. Vous pouvez effectuer la AWS Management Console mise à jour à partir de la dernière version de AWS CLI AWS CloudFormation, et `v0.164.0-rc.0` ou d'une `eksctl` version ultérieure. Vous pouvez avoir besoin de procéder ainsi pour fournir aux sous-réseaux davantage d'adresses IP disponibles afin de réussir la mise à niveau d'une version de cluster.	24 octobre 2023
Le rôle de cluster et le rôle de groupe de nœuds gérés prennent en charge les AWS Identity and Access Management politiques gérées par le client	Vous pouvez utiliser une stratégie IAM personnal isée sur le rôle de cluster, au lieu de la stratégie `AmazonEKSClusterPolicy` AWS gérée. Vous pouvez également utiliser une stratégie IAM personnal isée sur le rôle de nœud dans un groupe de nœuds gérés, au lieu de la stratégie `AmazonEKSWorkerNodePolicy` AWS gérée. Cela permet de créer une politique avec le moins de privilège s possible afin de répondre à des exigences strictes en matière de conformité.	23 octobre 2023

Correction du lien vers l'installation d'eksctl	Correction du lien d'installation d'eksctl après que la page ait été déplacée.	6 octobre 2023
Version préliminaire : support étendu d'Amazon EKS pour les versions de Kubernetes	Le support étendu des versions de Kubernetes vous permet de conserver une version spécifique de Kubernetes pendant plus de 14 mois.	4 octobre 2023
Supprimer les références à AWS App Mesh l'intégration	Intégrations d'Amazon EKS avec AWS App Mesh Remain pour les clients existants d'App Mesh uniquement.	29 septembre 2023
Kubernetes version `1.28`	Ajout de la prise en charge de la version `1.28` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	26 septembre 2023
Prise en charge de l'application de la politique réseau Kubernetes dans Amazon VPC CNI plugin for Kubernetes par les clusters existants	Vous pouvez utiliser la politique réseau Kubernetes dans les clusters existants avec le Amazon VPC CNI plugin for Kubernetes au lieu d'une solution tierce.	15 septembre 2023
Le module complémentaire Amazon EKS CoreDNS prend en charge la modification de la PDB	Vous pouvez modifier le `PodDisruptionBudget` du module complémentaire EKS pour CoreDNS dans les versions `v1.9.3-eksbuild.7` et ultérieures, et `v1.10.1-eksbuild.4` et ultérieures.	15 septembre 2023

Prise en charge d'Amazon EKS pour les sous-réseaux partagés	Nouvelles exigences et considérations relatives aux sous-réseaux partagéspour créer des clusters Amazon EKS dans des sous-réseaux partagés.	7 septembre 2023
Mises à jour de Qu'est-ce qu'Amazon EKS ?	Ajout de nouveaux cas d'utilisation courants et de nouvelles rubriques Architecture. Autres rubriques actualisées.	6 septembre 2023
Application de la stratégie réseau Kubernetes dans le Amazon VPC CNI plugin for Kubernetes	Vous pouvez utiliser la stratégie réseau Kubernetes avec le Amazon VPC CNI plugin for Kubernetes, au lieu d'avoir besoin d'une solution tierce.	29 août 2023
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible en Israël (Tel Aviv) (`il-central-1`) Région AWS.	1er août 2023
Stockage éphémère configurable pour Fargate	Vous avez la possibilité d'augmenter la capacité totale de stockage éphémère pour chaque Pod en cours d'exécution sur Amazon EKS Fargate.	31 juillet 2023
Prise en charge du module complémentaire pour le pilote CSI d'Amazon EFS	Vous pouvez désormais utiliser l'API AWS Management Console AWS CLI, et pour gérer le pilote Amazon EFS CSI.	26 juillet 2023

AWS mises à jour des politiques gérées - Nouvelle politique	Amazon EKS a ajouté une nouvelle politique AWS gérée.	26 juillet 2023
Kubernetesles mises à jour de version pour 1.27, 1.26, 1.25 et 1.24 sont désormais disponibles pour les clusters locaux sur AWS Outposts	Kubernetesles mises à jour des versions 1.27.3, 1.26.6, 1.25.11 et 1.24.15 sont désormais disponibles pour les clusters locaux sur AWS Outposts	20 juillet 2023
Prise en charge des préfixes IP pour les nœuds Windows	L'attribution de préfixes IP à vos nœuds peut vous permettre d'en héberger un nombre de Pods nettement plus élevé que lorsque vous attribuez des adresses IP secondaires individuelles à vos nœuds.	6 juillet 2023
Pilote CSI Amazon FSx pour OpenZFS	Vous pouvez désormais installer le pilote Amazon FSx pour OpenZFS CSI sur les clusters Amazon EKS.	30 juin 2023
Les Pods sur des nœuds Linux dans des clusters `IPv4` peuvent désormais communiquer avec des points de terminaison `IPv6`.	Après avoir attribué une adresse IPv6 à votre nœud, l'adresse `IPv4` de vos Pods est traduite en adresse réseau vers l'adresse `IPv6` du nœud sur lequel il s'exécute.	19 juin 2023

Windowsgroupes de nœuds gérés dans AWS GovCloud (US) Regions	Dans le AWS GovCloud (US) Regions, les groupes de nœuds gérés par Amazon EKS peuvent désormais exécuter Windows des conteneurs.	30 mai 2023
Kubernetes version `1.27`	Ajout de la prise en charge de la version `1.27` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	24 mai 2023
Kubernetes version `1.26`	Ajout de la prise en charge de la version `1.26` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	11 avril 2023
gMSA sans domaine	Vous pouvez désormais utiliser l'option gMSA sans domaine avec les Pods Windows.	27 mars 2023
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la région Asie-Pacifique (Melbourne) (`ap-southeast-4`) Région AWS.	10 mars 2023
Pilote CSI Amazon File Cache	Vous pouvez désormais installer le pilote CSI Amazon File Cache sur les clusters Amazon EKS.	3 mars 2023
Kubernetesla version 1.25 est désormais disponible pour les clusters locaux sur AWS Outposts	Vous pouvez désormais créer un cluster local Amazon EKS sur un Outpost en utilisant les versions `1.22` à `1.25` de Kubernetes.	1 mars 2023

Kubernetes version `1.25`	Ajout de la prise en charge de la version `1.25` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	22 février 2023
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	7 février 2023
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible en Asie-Pacifique (Hyderabad) (`ap-south-2`), en Europe (Zurich) (`eu-central-2`) et en Europe (Espagne) (`eu-south-2`). Régions AWS	6 février 2023
Kubernetesversions `1.21` — `1.24` sont désormais disponibles pour les clusters locaux sur AWS Outposts.	Vous pouvez désormais créer un cluster local Amazon EKS sur un Outpost en utilisant les versions `1.21` à `1.24` de Kubernetes. Auparavant, seule la version `1.21` était disponible.	17 janvier 2023
Amazon EKS prend désormais en charge AWS PrivateLink	Vous pouvez utiliser an AWS PrivateLink pour créer une connexion privée entre votre VPC et Amazon EKS.	16 décembre 2022
Prise en charge par Windows des groupes de nœuds gérés	Vous pouvez désormais utiliser Windows pour les groupes de nœuds gérés Amazon EKS.	15 décembre 2022

Des modules complémentaires Amazon EKS proposés par des fournisseurs indépendants de logiciels sont désormais disponibles dans AWS Marketplace	Vous pouvez désormais consulter et vous abonner aux modules complémentaires Amazon EKS proposés par des fournisseurs indépendants de logiciels par le biais de AWS Marketplace.	28 novembre 2022
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	17 novembre 2022
Kubernetes version `1.24`	Ajout de la prise en charge de la version `1.24` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	15 novembre 2022
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible au Moyen-Orient (Émirats arabes unis) (`me-central-1`) Région AWS.	3 novembre 2022
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	24 octobre 2022
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	20 octobre 2022
Les clusters locaux AWS Outposts sont désormais disponibles	Vous pouvez désormais créer un cluster local Amazon EKS sur un Outpost.	19 septembre 2022
Quotas basés sur les vCPU Fargate	Fargate effectue une transition des quotas basés sur les Pod vers des quotas basés sur les processeurs virtuels.	8 septembre 2022

AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	31 août 2022
Suivi des coûts	Amazon EKS prend désormais en charge Kubecost, qui vous permet de suivre les coûts ventilés par ressources Kubernetes, y compris les Pods, les nœuds, les espaces de noms et les étiquettes.	24 août 2022
AWS mises à jour des politiques gérées - Nouvelle politique	Amazon EKS a ajouté une nouvelle politique AWS gérée.	24 août 2022
AWS mises à jour des politiques gérées - Nouvelle politique	Amazon EKS a ajouté une nouvelle politique AWS gérée.	23 août 2022
Balisage des ressources pour la facturation	Ajout de la prise en charge de balises de répartition des coûts généré par `aws:eks:cluster-name` pour tous les clusters.	16 août 2022
Caractères génériques du profil de Fargate	Ajout de la prise en charge des caractères génériques du profil Fargate dans les critères de sélection pour les espaces de noms, les clés d'étiquette et les valeurs d'étiquette.	16 août 2022
Kubernetes version `1.23`	Ajout de la prise en charge de la version `1.23` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	11 août 2022

Consultez Kubernetes les ressources dans le AWS Management Console	Vous pouvez désormais afficher des informations sur les ressources Kubernetes déployées sur votre cluster à l'aide de la AWS Management Console.	3 mai 2022
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la région Asie-Pacifique (Jakarta) (`ap-southeast-3`) Région AWS.	2 mai 2022
Prise en charge de la page Observability et du module complémentaire ADOT	Ajout d'une page d'observa bilité et d' AWS une distribut ion pour OpenTelemetry (ADOT).	21 avril 2022
Kubernetes version `1.22`	Ajout de la prise en charge de la version `1.22` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	4 avril 2022
AWS mises à jour des politiques gérées - Nouvelle politique	Amazon EKS a ajouté une nouvelle politique AWS gérée.	4 avril 2022
Ajout de détails sur les correctifs de Pod Fargate	Lors de la mise à niveau des Pods Fargate, Amazon EKS essaie d'abord d'expulser les Pods en fonction des budgets d'interruption de votre Pod. Vous pouvez créer des règles d'événements pour réagir aux expulsions ayant échoué avant la suppression des Pods.	1er avril 2022

Version complète : prise en charge du module complémentaire pour le pilote CSI d'Amazon EBS	Vous pouvez désormais utiliser l'API AWS Management Console AWS CLI, et pour gérer le pilote Amazon EBS CSI.	31 mars 2022
AWS Outposts mise à jour du contenu	Instructions de déploiement d'un cluster Amazon EKS sur AWS Outposts.	22 mars 2022
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	21 mars 2022
Prise en charge de Windows `containerd`	Vous pouvez désormais sélectionner l'exécution `containerd` pour les nœuds Windows.	14 mars 2022
Ajout de considérations Amazon EKS Connector à la documentation de sécurité	Décrit le modèle de responsabilité partagée en ce qui concerne les clusters connectés.	25 février 2022
Attribuer des adresses `IPv6` à vos Pods et services	Vous pouvez désormais créer un cluster de version `1.21` ou ultérieure qui attribue des adresses `IPv6` à vos Pods et services.	6 janvier 2022
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	Amazon EKS a mis à jour une politique AWS gérée existante.	13 décembre 2021

Version préliminaire : prise en charge du module complémentaire pour le pilote CSI d'Amazon EBS	Vous pouvez désormais prévisualiser à l'aide de l'API AWS Management Console AWS CLI, et pour gérer le pilote Amazon EBS CSI.	9 décembre 2021
Prise en charge de Karpenter Autoscaler	Vous pouvez désormais utiliser le projet open source Karpenter pour mettre à l'échelle automatique vos nœuds.	29 novembre 2021
Prise en charge du filtre Kubernetes Fluent Bit dans la journalisation Fargate	Vous pouvez désormais utiliser le filtre Kubernetes Fluent Bit avec la journalisation Fargate.	10 novembre 2021
Prise en charge de Windows disponible dans le plan de contrôle	La prise en charge de Windows est désormais disponible dans votre plan de contrôle. Vous n'avez plus besoin de l'activer dans votre plan de données.	9 novembre 2021
Bottlerocket ajouté en tant que type d'AMI pour les groupes de nœuds gérés	Auparavant, Bottlerocket était uniquement disponible en tant qu'option de nœud autogéré. Il peut désormais être configuré comme un groupe de nœuds gérés, ce qui réduit l'effort nécessaire pour répondre aux exigences de conformité des nœuds.	28 octobre 2021

Prise en charge du pilote DL1	Les AMI Amazon Linux personnalisées prennent désormais en charge les charges de travail de Deep Learning pour Amazon Linux 2. Cette activation permet une configuration de base générique sur site ou dans le cloud.	25 octobre 2021
Prise en charge de la vidéo VT1	Les AMI Amazon Linux personnalisées prennent maintenant en charge VT1 pour certaines distributions. Cette mise en œuvre annonce les dispositifs Xilinx U30 sur votre cluster Amazon EKS.	13 septembre 2021
Amazon EKS Connector est désormais disponible	Vous pouvez utiliser le connecteur Amazon EKS pour enregistrer et connecter n'importe quel Kubernetes cluster conforme AWS et le visualiser dans la console Amazon EKS.	8 septembre 2021
Amazon EKS Anywhere est désormais disponible	Amazon EKS Anywhere est une nouvelle option de déploiement pour Amazon EKS que vous pouvez utiliser pour créer et exploiter facilement des clusters Kubernetes sur site.	8 septembre 2021

Pilote Amazon FSx pour NetApp ONTAP CSI	Ajout d'une rubrique qui résume le pilote Amazon FSx NetApp pour ONTAP CSI et fournit des liens vers d'autres références.	2 septembre 2021
Les groupes de nœuds gérés calculent désormais automatiquement le nombre maximal de Pods recommandés par Amazon EKS pour les nœuds	Les groupes de nœuds gérés calculent désormais automatiquement le nombre maximal de Pods Amazon EKS pour les nœuds que vous déployez sans modèle de lancement ou avec un modèle de lancement dans lequel vous n'avez pas spécifié d'ID d'AMI.	30 août 2021
Supprimer la gestion Amazon EKS des paramètres du module complémentaire sans supprimer le logiciel complémentaire Amazon EKS	Vous pouvez désormais supprimer un module complémentaire Amazon EKS sans supprimer le logiciel complémentaire de votre cluster.	20 août 2021
Créer des Pods multi-résidents à l'aide de Multus	Vous pouvez maintenant ajouter plusieurs interfaces réseau à un Pod à l'aide de Multus.	2 août 2021
Ajouter d'autres adresses IP à vos nœuds Linux Amazon EC2	Vous pouvez maintenant ajouter beaucoup plus d'adresses IP à vos nœuds Linux Amazon EC2. Cela signifie que vous pouvez exécuter une densité plus élevée de Pods sur chaque nœud.	27 Juillet 2021

Amorçage d'exécution `containerd`	L'Amazon Machine Image (AMI) Linux Amazon accélérée optimisée pour Amazon EKS contient désormais un indicateur d'amorçag e que vous pouvez utiliser pour activer l'environnement d'exécution `containerd` dans les AMI optimisées pour Amazon EKS et les AMI Bottlerocket. Cet indicateur est disponible dans toutes les versions de Kubernetes prises en charge de l'AMI.	19 juillet 2021
Kubernetes version `1.21`	Ajout de la prise en charge de la version `1.21` de Kubernete s.	19 juillet 2021
Ajout d'une rubrique de politiques gérées	Liste de toutes les politiques gérées par Amazon EKS IAM et des modifications qui leur ont été apportées depuis le 17 juin 2021.	17 juin 2021
Utiliser des groupes de sécurité pour les Pods avec Fargate	Vous pouvez désormais utiliser des groupes de sécurité pour les Pods avec Fargate, en plus de les utiliser avec des nœuds Amazon EC2.	1er juin 2021

Ajout des modules complémentaires Amazon EKS CoreDNS et `kube-proxy`	Amazon EKS peut désormais vous aider à gérer les CoreDNS et les modules complémentaires Amazon EKS `kube-proxy` pour votre cluster.	19 mai 2021
Kubernetes version `1.20`	Ajout de la prise en charge de la version `1.20` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	18 mai 2021
AWS Load Balancer Controller `r2.2.0` lancé	Vous pouvez désormais utiliser le AWS Load Balancer Controller pour créer des Elastic Load Balancers à l'aide d'instances ou de cibles IP.	14 mai 2021
Taints de noeuds pour les groupes de nœuds gérés	Amazon EKS prend désormais en charge l'ajout de taints de notes aux groupes de nœuds gérés.	11 mai 2021
Chiffrement des secrets pour les clusters existants	Amazon EKS prend désormais en charge l'ajout du chiffrement des secrets aux clusters existants.	26 février 2021
Kubernetes version `1.19`	Ajout de la prise en charge de la version `1.19` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	16 février 2021

Amazon EKS prend désormais en charge les fournisseurs d'identité OpenID Connect (OIDC) comme méthode d'authentification des utilisateurs d'un cluster version `1.16` ou ultérieure.	Les fournisseurs d'identité OIDC peuvent être utilisés avec ou comme alternative à AWS Identity and Access Management (IAM).	12 février 2021
Afficher les ressources des nœuds et des charges de travail dans le AWS Management Console	Vous pouvez désormais afficher les détails sur vos nœuds gérés, autogérés et Fargate, ainsi que sur vos charges de travail Kubernetes déployées dans la AWS Management Console.	1er décembre 2020
Déployer des types d'instances Spot dans un groupe de nœuds gérés	Vous pouvez désormais déployer plusieurs types d'instances Spot ou à la demande dans un groupe de nœuds gérés.	1er décembre 2020
Amazon EKS peut désormais gérer des modules complémentaires spécifiques pour votre cluster	Vous pouvez gérer vous-même les modules complémentaires ou autoriser Amazon EKS à contrôler le lancement et la version d'un module complémentaire via l'API Amazon EKS.	1er décembre 2020
Partager un ALB sur plusieurs entrées	Vous pouvez désormais partager un AWS Application Load Balancer (ALB) sur plusieurs entrées. Kubernetes Auparavant, vous deviez déployer un ALB distinct pour chaque entrée.	23 octobre 2020

Prise en charge des cibles IP NLB	Vous pouvez désormais déployer un Network Load Balancer avec des cibles IP. Cela signifie que vous pouvez utiliser un NLB pour équilibrer la charge du trafic réseau vers les Pods Fargate et directement vers les Pods qui s'exécutent sur les nœuds Amazon EC2.	23 octobre 2020
Kubernetes version `1.18`	Ajout de la prise en charge de la version `1.18` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	13 octobre 2020
Spécifiez un bloc CIDR personnalisé pour l'attribution de l'adresse IP du service Kubernetes.	Vous pouvez désormais spécifier un bloc CIDR personnalisé à partir duquel Kubernetes attribue les adresses IP de service.	29 septembre 2020
Attribuer des groupes de sécurité à des Pods individuels	Vous pouvez maintenant associer différents groupes de sécurité à certains des Pods individuels qui s'exécutent sur de nombreux types d'instances Amazon EC2.	9 septembre 2020
Déployer Bottlerocket sur vos nœuds	Vous pouvez désormais déployer des nœuds qui exécutent Bottlerocket.	31 août 2020
La possibilité de lancer des nœuds Arm est généralement disponible	Vous pouvez maintenant lancer des nœuds Arm dans des groupes de nœuds gérés et autogérés.	17 août 2020

Modèles de lancement de groupes de nœuds gérés et AMI personnalisée	Vous pouvez maintenant déployer un groupe de nœuds géré qui utilise un modèle de lancement Amazon EC2. Le modèle de lancement peut spécifier une AMI personnal isée, si vous le souhaitez.	17 août 2020
Support EFS pour AWS Fargate	Vous pouvez désormais utiliser Amazon EFS avec AWS Fargate.	17 août 2020
Mise à jour de la version de plateforme Amazon EKS	Il s'agit d'une nouvelle version de plateforme qui comporte des correctifs et des améliorat ions de sécurité. Cela inclut la prise en charge UDP pour les services de type `LoadBalan cer` lorsque vous utilisez Network Load Balancers avec la version `1.15` de Kubernetes ou une version ultérieure. Pour plus d'informations, consultez le problème Allow UDP for AWS Network Load Balancer sur. GitHub	12 août 2020
Région AWS Expansion d'Amazon EKS	Amazon EKS est maintenan t disponible dans les Régions AWS Afrique (Le Cap) (`af-south-1`) et Europe (Milan) (`eu-south-1`).	6 août 2020

Métriques d'utilisation de Fargate	AWS Fargate fournit des statistiques CloudWatch d'utilisation qui fournissent une visibilité sur l'utilisation des ressources Fargate On-Demand par votre compte.	3 août 2020
Kubernetes version `1.17`	Ajout de la prise en charge de la version `1.17` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	10 juillet 2020
Créez et gérez des ressources App Mesh depuis Kubernetes avec le contrôleur App Mesh pour Kubernetes	Vous pouvez créer et gérer des ressources App Mesh depuis Kubernetes. Le contrôleur injecte également automatiquement le proxy Envoy et les conteneurs init dans les Pods que vous déployez.	18 juin 2020
Amazon EKS prend désormais en charge les nœuds Inf1 Amazon EC2	Vous pouvez ajouter des nœuds Amazon EC2 Inf1 à votre cluster.	4 juin 2020
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans les pays AWS GovCloud (USA Est) (`us-gov-east-1`) et AWS GovCloud (USA Ouest) (`us-gov-west-1`). Régions AWS	13 mai 2020

Kubernetes1.12 n'est plus pris en charge sur Amazon EKS	La version 1.12 de Kubernetes n'est plus prise en charge sur Amazon EKS. Veuillez mettre à jour les clusters 1.12 vers la version 1.13 ou une version ultérieure pour éviter toute interruption de service.	12 mai 2020
Kubernetes version 1.16	Ajout de la prise en charge de la version 1.16 de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	30 avril 2020
Ajout du rôle AWSServiceRoleForAmazonEKSlié au service	Le rôle AWSServiceRoleForAmazonEKSlié au service a été ajouté.	16 avril 2020
Kubernetes version 1.15	Ajout de la prise en charge de la version 1.15 de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	10 mars 2020
Région AWS Expansion d'Amazon EKS	Amazon EKS est maintenant disponible dans les Régions AWS Beijing (cn-north-1) et Ningxia (cn-northwest-1).	26 février 2020
Pilote CSI FSx for Lustre	Ajout d'une rubrique pour l'installation du pilote CSI FSx for Lustre sur les clusters Amazon EKS Kubernetes 1.14.	23 décembre 2019

Restreindre l'accès réseau au point de terminaison d'accès public d'un cluster	Avec cette mise à jour, vous pouvez utiliser Amazon EKS pour restreindre les plages CIDR qui peuvent communiquer avec le point de terminaison d'accès public du serveur API Kubernetes.	20 décembre 2019
Résoudre l'adresse du point de terminaison de l'accès privé pour un cluster depuis l'extérieur d'un VPC	Avec cette mise à jour, vous pouvez utiliser Amazon EKS pour résoudre le point de terminaison d'accès privé du serveur API Kubernetes depuis l'extérieur d'un VPC.	13 décembre 2019
(Bêta) Nœuds d'instance Amazon EC2 A1 Amazon EC2	Lancez les nœuds d'instance Amazon EC2 Amazon EC2 A1 qui s'enregistrent dans votre cluster Amazon EKS.	4 décembre 2019
Création d'un cluster sur AWS Outposts	Amazon EKS prend désormais en charge la création de clusters sur AWS Outposts.	3 décembre 2019
AWS Fargate sur Amazon EKS	Les clusters Amazon EKS Kubernetes prennent désormais en charge les Pods exécutés sur Fargate.	3 décembre 2019
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible au Canada (Centre) (`ca-central-1`) Région AWS.	21 novembre 2019

Groupes de nœuds gérés	Les groupes de nœuds gérés Amazon EKS automatisent l'approvisionnement et la gestion du cycle de vie des nœuds (instances Amazon EC2) pour les clusters Amazon EKS Kubernetes.	18 novembre 2019
Mise à jour de la version de plateforme Amazon EKS	Nouvelles versions de plateforme pour résoudre `CVE-2019-11253` .	6 novembre 2019
Kubernetes`1.11` n'est plus pris en charge sur Amazon EKS	La version `1.11` de Kubernetes n'est plus prise en charge sur Amazon EKS. Veuillez mettre à jour les clusters `1.11` vers la version `1.12` ou ultérieure pour éviter toute interruption de service.	4 novembre 2019
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la Région AWS Amérique du Sud (Sao Paulo) (`sa-east-1`).	16 octobre 2019
Windows Prise en charge de	Les clusters Amazon EKS exécutant la version Kubernetes de `1.14` prennent désormais en charge les charges de travail Windows.	7 octobre 2019

Autoscaling	Ajout d'un chapitre pour couvrir certains des différents types de scalabilité automatiq ue Kubernetes pris en charge dans les clusters Amazon EKS.	30 septembre 2019
Mise à jour du tableau de bord Kubernetes	Rubrique mise à jour pour l'installation du tableau de bord Kubernetes dans les clusters Amazon EKS afin d'utiliser la version bêta `2.0`.	28 septembre 2019
Pilote CSI Amazon EFS	Ajout d'une rubrique pour l'installation du pilote CSI Amazon EFS sur les clusters Amazon EKS Kubernete s `1.14`.	19 septembre 2019
Paramètre Amazon EC2 Systems Manager pour l'ID d'AMI optimisée pour Amazon EKS	Ajout d'une rubrique pour la récupération de l'ID d'AMI optimisée pour Amazon EKS à l'aide d'un Amazon EC2 Systems Manager. Le paramètre vous évite de devoir rechercher les ID d'AMI.	18 septembre 2019
Étiquetage des ressources Amazon EKS	Vous pouvez gérer l'identif ication de vos clusters Amazon EKS.	16 septembre 2019
Pilote CSI Amazon EBS	Ajout d'une rubrique pour l'installation du pilote CSI Amazon EBS sur les clusters Amazon EKS Kubernete s `1.14`.	9 septembre 2019

Nouvelle AMI optimisée pour Amazon EKS corrigée pour CVE-2019-9512 et CVE-2019-9514	Amazon EKS a mis à jour l'AMI optimisée pour Amazon EKS pour résoudre CVE-2019-9512 et CVE-2019-9514 .	6 septembre 2019
Annonce de l'obsolescence de Kubernetes1.11 dans Amazon EKS	Amazon EKS a cessé de prendre en charge la version 1.11 de Kubernetes le 4 novembre 2019.	4 septembre 2019
Kubernetes version 1.14	Ajout de la prise en charge de la version 1.14 de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	3 septembre 2019
Rôles IAM pour les comptes de service	Avec les rôles IAM pour les comptes de service sur les clusters Amazon EKS, vous pouvez associer un rôle IAM à un compte de service Kubernetes. Grâce à cette fonction, vous n'avez plus besoin de fournir des autorisations étendues au rôle IAM du nœud. De cette façon, Pods sur ce nœud, vous pouvez appeler AWS des API.	3 septembre 2019
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la Région AWS Moyen-Orient (Bahreïn) (me-south-1).	29 août 2019

Mise à jour de la version de plateforme Amazon EKS	Nouvelles versions de plateforme pour résoudre `CVE-2019-9512` et `CVE-2019-9514`.	28 août 2019
Mise à jour de la version de plateforme Amazon EKS	Nouvelles versions de plateforme pour résoudre `CVE-2019-11247` et `CVE-2019-11249`.	5 août 2019
Extension de région Amazon EKS	Amazon EKS est désormais disponible dans la Région AWS Asie-Pacifique (Hong Kong) (`ap-east-1`).	31 juillet 2019
Kubernetes`1.10` n'est plus pris en charge sur Amazon EKS	La version `1.10` de Kubernetes n'est plus prise en charge sur Amazon EKS. Mettez à jour tous les clusters `1.10` vers la version `1.11` ou une version ultérieure pour éviter toute interruption de service.	30 juillet 2019
Ajout d'une rubrique sur le contrôleur d'entrée ALB	Le contrôleur d'entrée AWS ALB pour Kubernetes est un contrôleur qui provoque la création d'un ALB lors de la création de ressources d'entrée.	11 juillet 2019
Nouvelle AMI optimisée pour Amazon EKS	Suppression du fichier binaire `kubectl` inutile des AMI.	3 juillet 2019
Kubernetes version `1.13`	Ajout de la prise en charge de la version `1.13` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	18 juin 2019

Nouvelle AMI optimisée pour Amazon EKS corrigée pour `AWS-2019-005`	Amazon EKS a mis à jour l'AMI optimisée pour Amazon EKS pour résoudre les vulnérabilités décrites dans `AWS-2019-005` .	17 juin 2019
Annonce de l'arrêt de la prise en charge de Kubernetes`1.10` dans Amazon EKS	Amazon EKS a cessé de prendre en charge la version `1.10` de Kubernetes le 22 juillet 2019.	21 mai 2019
Mise à jour de la version de plateforme Amazon EKS	Nouvelle version de la plateforme pour les clusters Kubernetes `1.11` et `1.10`, afin de prendre en charge les noms DNS personnalisés dans le certificat `kubelet` et d'améliorer les performances de `etcd`.	21 mai 2019

Commande AWS CLIget-token de la	La commande aws eks get-token a été ajoutée à la AWS CLI. Vous n'avez plus besoin d'installer l'authentificateur AWS IAM pour créer des jetons de sécurité client Kubernetes pour les communications avec le serveur d'API du cluster. Mettez à niveau votre AWS CLI installation vers la dernière version pour utiliser cette nouvelle fonctionnalité. Pour plus d'informations, consultez Installation d' AWS Command Line Interface dans le Guide de l'utilisateur AWS Command Line Interface .	10 mai 2019
Démarrer avec eksctl	Ce guide de démarrage explique comment installer toutes les ressources nécessaires pour démarrer avec Amazon EKS en utilisant eksctl. Il s'agit d'un utilitaire de ligne de commande simple pour créer et gérer les clusters Kubernetes sur Amazon EKS.	10 mai 2019

Mise à jour de la version de plateforme Amazon EKS	Nouvelle version de plateforme pour les clusters Kubernetes `1.12`, afin que ceux-ci prennent en charge les noms DNS personnalisés dans le certificat `kubelet` et améliorer les performances `etcd`. Cela corrige un bogue qui faisait que les démons `kubelet` des nœuds redemandaient un nouveau certificat au bout de quelques secondes.	8 mai 2019
Didacticiel Prometheus	Ajout d'une rubrique pour le déploiement de Prometheus dans votre cluster Amazon EKS.	5 avril 2019
Journalisation de plan de contrôle d'Amazon EKS	Avec cette mise à jour, vous pouvez obtenir des journaux d'audit et de diagnostic directement à partir du volet de contrôle d'Amazon EKS. Vous pouvez utiliser ces CloudWatch journaux dans votre compte comme référence pour sécuriser et exécuter des clusters.	4 avril 2019
Kubernetes version `1.12`	Ajout de la prise en charge de la version `1.12` de Kubernetes pour les nouveaux clusters et les mises à niveau de version.	28 mars 2019

Ajout du guide de démarrage App Mesh	Ajout de la documentation pour démarrer avec App Mesh et Kubernetes.	27 mars 2019
Accès privé au point de terminaison du serveur d'API Amazon EKS	Ajout de la documentation sur la désactivation de l'accès public pour le point de terminaison du serveur API Kubernetes de votre cluster Amazon EKS.	19 mars 2019
Ajout d'une rubrique pour l'installation du serveur de métriques Kubernetes	Le serveur de métriques Kubernetes est un agrégateur de données sur l'utilisation des ressources dans votre cluster.	18 mars 2019
Ajout d'une liste de projets open source connexes	Ces projets open source étendent les fonctionnalités des Kubernetes clusters exécutés sur AWS, y compris les clusters gérés par Amazon EKS.	15 mars 2019
Ajout d'une rubrique pour l'installation de Helm en local	Le gestionnaire de package `helm` pour Kubernetes vous permet d'installer et de gérer des applications dans votre cluster Kubernetes. Cette rubrique explique comment installer et exécuter localement les fichiers binaires `helm` et `tiller`. Vous pouvez ainsi installer et gérer des graphiques à l'aide de la CLI Helm sur votre système local.	11 mars 2019

Mise à jour de la version de plateforme Amazon EKS	Nouvelle version de la plateforme qui met à jour les clusters Amazon EKS Kubernetes `1.11` vers le niveau de correctif `1.11.8` pour résoudre `CVE-2019-1002100` .	8 mars 2019
Augmentation de la limite de clusters	Amazon EKS a augmenté de 3 à 50 le nombre de clusters que vous pouvez créer dans une Région AWS .	13 février 2019
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible en Europe (Londres) (`eu-west-2`), en Europe (Pari`eu-west-3`) () et en Asie-Pacifique (Mumbai) (`ap-south-1`) Régions AWS.	13 février 2019
Nouvelle AMI optimisée pour Amazon EKS corrigée pour `ALAS-2019-1156`	Amazon EKS a mis à jour l'AMI optimisée pour Amazon EKS afin de résoudre la vulnérabilité décrite dans `ALAS-2019-1156` .	11 février 2019
Nouvelle AMI optimisée pour Amazon EKS corrigée pour `ALAS2-2019-1141`	Amazon EKS a mis à jour l'AMI optimisée pour Amazon EKS pour résoudre les vulnérabilités et expositions courantes (CVE) référencées dans `ALAS2-2019-1141` .	9 janvier 2019

Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la région Asie-Pacifique (Séoul) (`ap-northeast-2`) Région AWS.	9 janvier 2019
Extension de région Amazon EKS	Amazon EKS est désormais disponible dans les pays supplémentaires suivants Régions AWS : Europe (Francfort`eu-central-1`) (), Asie-Pacifique (Tokyo`ap-northeast-1`) (), Asie-Pacifique (Singapour`ap-southeast-1`) () et Asie-Pacifique (Sydney) (`ap-southeast-2`).	19 décembre 2018
Mise à jour du cluster Amazon EKS	Ajout de documentation relative à la mise à jour de la version de Kubernetes des clusters et au remplacement de nœuds pour Amazon EKS.	12 décembre 2018
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la Région AWS Europe (Stockholm) (`eu-north-1`).	11 décembre 2018
Mise à jour de la version de plateforme Amazon EKS	Nouvelle version de la plateforme mettant à jour les correctifs de Kubernetes à la version `1.10.11` pour résoudre `CVE-2018-1002105` .	4 décembre 2018

Ajout de la prise en charge de la version `1.0.0` du contrôleur d'entrée ALB	Le contrôleur d'entrée ALB publie une version `1.0.0` avec le support formel de. AWS	20 novembre 2018
Ajout de la prise en charge de la configuration réseau CNI	La version `1.2.1` du Amazon VPC CNI plugin for Kubernetes prend désormais en charge la configuration réseau personnalisée pour les interfaces réseau de Pod secondaires.	16 octobre 2018
Ajout de la prise en charge de `MutatingAdmissionWebhook` et `ValidatingAdmissionWebhook`	La version `1.10-eks.2` de la plateforme Amazon EKS prend désormais en charge les contrôleurs d'admission `MutatingAdmissionWebhook` et `ValidatingAdmissionWebhook` .	10 octobre 2018
Ajout d'informations d'AMI de partenaire	Canonical a établi un partenariat avec Amazon EKS pour créer des AMI de nœuds que vous pouvez utiliser dans vos clusters.	3 octobre 2018
Ajout d'instructions pour la commande `update-kubeconfig` de l' AWS CLI	Amazon EKS a ajouté le `update-kubeconfig` au afin AWS CLI de simplifier le processus de création d'un `kubeconfig` fichier permettant d'accéder à votre cluster.	21 septembre 2018

Nouvelles AMI optimisées pour Amazon EKS	Amazon EKS a mis à jour les AMI optimisées pour Amazon EKS (avec et sans prise en charge GPU) pour fournir divers correctifs de sécurité et optimisations d'AMI.	13 septembre 2018
Région AWS Expansion d'Amazon EKS	Amazon EKS est désormais disponible dans la région Europe (Irlande) (`eu-west-1`).	5 septembre 2018
Mise à jour de la version de plateforme Amazon EKS	Nouvelle version de plateforme avec support de la couche d'agrégation et Horizontal Pod Autoscaler (HPA) de Kubernetes.	31 août 2018
Nouvelles AMI optimisées pour Amazon EKS et prise en charge GPU	Amazon EKS a mis à jour l'AMI optimisée Amazon EKS afin d'utiliser un nouveau modèle de AWS CloudFormation nœud et un nouveau script d'amorçage. En outre, une nouvelle AMI optimisée pour Amazon EKS avec prise en charge GPU est disponible.	22 août 2018
Nouvelle AMI optimisée pour Amazon EKS corrigée pour `ALAS2-2018-1058`	Amazon EKS a mis à jour l'AMI optimisée pour Amazon EKS pour résoudre les vulnérabilités et expositions courantes (CVE) référencées dans `ALAS2-2018-1058`.	14 août 2018

Scripts de génération d'AMI optimisée pour Amazon EKS	Amazon EKS comporte des scripts de génération open source qui sont utilisés pour créer l'AMI optimisée pour Amazon EKS. Ces scripts de génération sont maintenant disponibles sur GitHub.	10 juillet 2018
Version initiale d'Amazon EKS	Documentation initiale pour le lancement de service	5 juin 2018

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.