aws

Guide du développeur

AWS Key Management Service

AWS Key Management Service: Guide du développeur

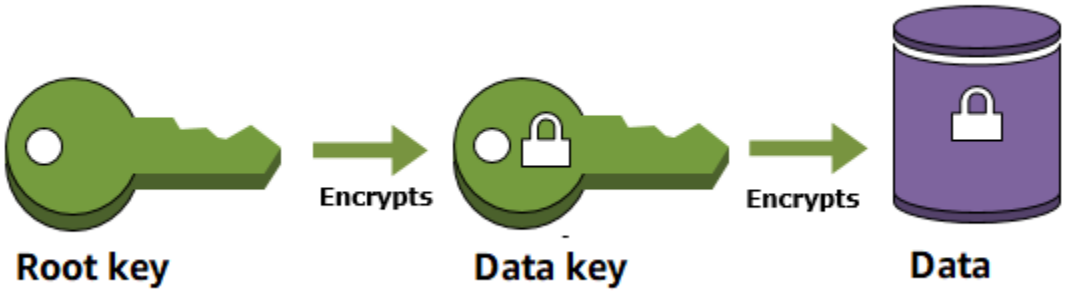
Table of Contents

AWS Key Management Service

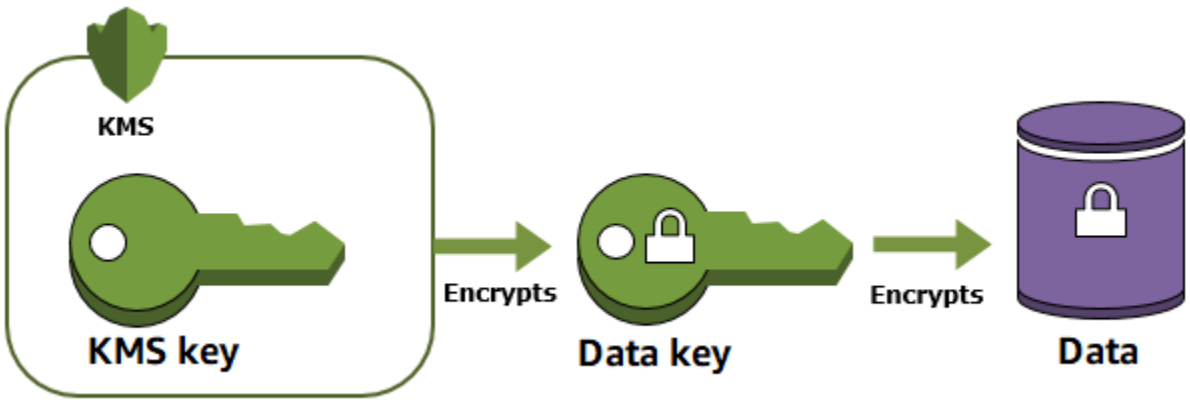
AWS Key Management Service (AWS KMS) est un service AWS géré qui vous permet de créer et de contrôler facilement les clés de chiffrement utilisées pour chiffrer vos données. Les éléments AWS KMS keys que vous créez dans AWS KMS sont protégés par des modules de sécurité matériels (HSM) validés par la norme de sécurité FIPS 140-3 de niveau 3. Ils ne partent jamais AWS KMS non chiffrés. Pour utiliser ou gérer vos clés KMS, vous interagissez avec AWS KMS.

Pourquoi utiliser AWS KMS ?

Lorsque vous chiffrez des données, vous devez protéger votre clé de chiffrement. Si vous chiffrez votre clé, vous devez protéger sa clé de chiffrement. Enfin, vous devez protéger la clé de chiffrement de niveau le plus élevé (appelée clé racine) de la hiérarchie qui protège vos données. C'est là qu' AWS KMS intervient.



AWS KMS protège vos clés root. Les clés KMS sont créées, gérées, utilisées et supprimées dans leur intégralité AWS KMS. Ils ne quittent jamais le service sans être chiffrés. Pour utiliser ou gérer vos clés KMS, vous devez appeler AWS KMS.



En outre, vous pouvez créer et gérer des politiques clés dans AWS KMS, en veillant à ce que seuls les utilisateurs de confiance aient accès aux clés KMS.

AWS KMS dans Régions AWS

Les éléments pris Régions AWS en charge AWS KMS sont répertoriés dans AWS Key Management Service Endpoints et Quotas. Si une AWS KMS fonctionnalité n'est pas prise en charge par une Région AWS fonctionnalité AWS KMS compatible, la différence régionale est décrite dans la rubrique consacrée à la fonctionnalité.

AWS KMS tarification

Comme pour les autres AWS produits, leur utilisation AWS KMS ne nécessite pas de contrats ou d'achats minimaux. Pour plus d'informations sur la AWS KMS tarification, consultez la section AWS Key Management Service Tarification.

AWS KMS accord de niveau de service

AWS Key Management Service est soutenu par un accord de niveau de service qui définit notre politique de disponibilité des services.

Accès AWS Key Management Service

Vous pouvez travailler avec AWS KMS les méthodes suivantes :

AWS Management Console

La console est une interface utilisateur basée sur le Web pour la gestion AWS KMS et les AWS ressources. Si vous vous êtes inscrit à un Compte AWS, vous pouvez accéder à la AWS KMS console en vous connectant à la page d'accueil AWS Management Console et en choisissant sur la page AWS KMS d' AWS Management Console accueil.

Autorisations requises pour utiliser la AWS KMS console

Pour utiliser la AWS KMS console, les utilisateurs doivent disposer d'un ensemble minimal d'autorisations leur permettant de travailler avec les AWS KMS ressources qu'ils contiennent Compte AWS. Outre ces autorisations AWS KMS , les utilisateurs doivent également être autorisés à répertorier les utilisateurs et rôles IAM. Si vous créez une politique IAM plus restrictive que les autorisations minimales requises, la console AWS KMS ne fonctionnera pas comme prévu pour les utilisateurs dotés de cette politique IAM.

Pour connaître les autorisations minimales requises pour accorder à un utilisateur l'accès en lecture seule à la console AWS KMS , consultez Autoriser un utilisateur à afficher les clés KMS dans la AWS KMS console.

Pour permettre aux utilisateurs d'utiliser la AWS KMS console pour créer et gérer des clés KMS, associez la politique AWSKeyManagementServicePowerUsergérée à l'utilisateur, comme décrit dansAWS politiques gérées pour AWS Key Management Service.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui utilisent l' AWS KMS API via le AWS SDKsAWS Command Line Interface, ou Outils AWS pour PowerShell. Cependant, vous devez accorder à ces utilisateurs l'autorisation d'utiliser l'API. Pour de plus amples informations, veuillez consulter Référence des autorisations .

AWS Command Line Interface

Vous pouvez utiliser les AWS CLI outils pour émettre des commandes ou créer des scripts sur la ligne de commande de votre système afin d'effectuer AWS (y compris AWS KMS) des tâches.

Pour plus d'informations sur l'utilisation AWS KMS via le AWS CLI, consultez le manuel de référence des AWS CLI commandes

AWS KMS REST API

L'architecture de AWS KMS est conçue pour être indépendante du langage de programmation, en utilisant des interfaces AWS compatibles pour stocker et récupérer des objets. Vous pouvez accéder à S3 et par AWS programmation en utilisant le. AWS KMS REST API REST APIIl s'agit d'une interface HTTP pour AWS KMS. Avec leREST API, vous utilisez des requêtes HTTP standard pour créer, récupérer et supprimer des compartiments et des objets.

Pour plus d'informations sur l'utilisation du AWS KMS REST API, consultez la référence de AWS Key Management Service l'API

AWS SDKs

AWS fournit SDKs (kits de développement logiciel) composés de bibliothèques et d'exemples de code pour les langages de programmation et les plateformes courants (Java JavaScript, C, Python, etc.). Ils AWS SDKs fournissent un moyen pratique de créer un accès programmatique à AWS KMS et AWS. AWS KMS est un REST service. Vous pouvez envoyer des demandes à AWS KMS l'aide des bibliothèques du AWS SDK, qui encapsulent le sous-jacent AWS KMS REST API et simplifient vos tâches de programmation. Pour plus d'informations sur les AWS SDKs, notamment sur la manière de les télécharger et de les installer, consultez la section Outils sur lesquels vous pouvez vous appuyer AWS.

C'Exemples de code pour AWS KMS l'utilisation AWS SDKsest un bon point de départ pour AWS KMS utiliser le AWS SDKs.

Utilisation de ce service avec un AWS SDK

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

Documentation SDK	Exemples de code
AWS SDK pour C++	AWS SDK pour C++ exemples de code

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

Exemple de disponibilité

Vous n’avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien Faire un commentaire en bas de cette page.

AWS Encryption SDK

AWS Encryption SDK Il s'agit d'un outil permettant d'implémenter le chiffrement côté client dans votre application. Il ne fournit pas un accès complet au KMS, mais il s'intègre AWS KMS au SDK

autonome ou peut être utilisé en tant que tel sans faire référence aux clés KMS. Des bibliothèques sont disponibles pour Java JavaScript, C, Python et d'autres langages de programmation.

Pour plus d'informations, consultez le Manuel du développeur AWS Encryption SDK.

AWS KMS key politiques et politiques IAM

AWS KMS cohérence éventuelle

L' AWS KMS API suit un modèle de cohérence éventuel en raison de la nature distribuée du système. Par conséquent, les modifications apportées aux AWS KMS ressources risquent de ne pas être immédiatement visibles pour les commandes suivantes que vous exécuterez.

Lorsque vous effectuez des appels AWS KMS d'API, il se peut qu'il y ait un bref délai avant que la modification ne soit disponible dans son intégralité AWS KMS. La propagation de la modification dans l'ensemble du système prend généralement moins de quelques secondes, mais dans certains cas, cela peut prendre plusieurs minutes. Vous risquez de recevoir des erreurs inattendues, telles qu'un `NotFoundException` ou un `InvalidStateException`, pendant cette période. Par exemple, AWS KMS peut renvoyer un `NotFoundException` si vous appelez `GetParametersForImport` immédiatement après avoir appelé`CreateKey`.

Nous vous recommandons de configurer une stratégie de relance pour vos AWS KMS clients afin de relancer automatiquement les opérations après une brève période d'attente. Pour plus d'informations, consultez la section Comportement des tentatives dans le guide de référence AWS SDKs et Tools.

Pour les appels d'API liés à un octroi, vous pouvez utiliser un jeton d'octroi pour éviter tout retard potentiel et utiliser immédiatement les autorisations d'un octroi. Pour plus d'informations, consultez la rubrique relative à la cohérence à terme (pour les octrois).

Utilisation du TLS post-quantique hybride avec AWS KMS

AWS Key Management Service (AWS KMS) prend en charge une option hybride d'échange de clés post-quantique pour le protocole de chiffrement réseau TLS (Transport Layer Security). Vous pouvez utiliser cette option TLS lorsque vous vous connectez aux points de terminaison de l'API AWS KMS . Ces fonctionnalités optionnelles d'échange de clés post-quantiques hybrides sont au moins aussi sécurisées que le chiffrement TLS que nous utilisons aujourd'hui et sont susceptibles d'offrir des avantages supplémentaires en matière de sécurité à long terme. Cependant, elles affectent la latence et le débit par rapport aux protocoles d'échange de clés classiques utilisés aujourd'hui.

Les données que vous envoyez à AWS Key Management Service (AWS KMS) sont protégées en transit par le cryptage fourni par une connexion TLS (Transport Layer Security). Les suites de chiffrement classiques que AWS KMS prend en charge pour les sessions TLS rendent les attaques de force brute contre les mécanismes d'échange clés irréalisables avec la technologie actuelle. Cependant, si l'informatique quantique à grande échelle devient courante à l'avenir, les suites de chiffrement classiques utilisées dans les mécanismes d'échange de clés TLS seront sensibles à ces attaques. Si vous développez des applications qui reposent sur la confidentialité à long terme des données transmises via une connexion TLS, vous devriez envisager de migrer vers la cryptographie post-quantique avant que des ordinateurs quantiques à grande échelle ne soient disponibles. AWS travaille à préparer ce futur, et nous voulons que vous soyez également bien préparés.

Afin de protéger les données chiffrées aujourd'hui contre d'éventuelles attaques futures, AWS participe avec la communauté cryptographique au développement d'algorithmes résistants aux quanta ou post-quantiques. Nous avons mis en œuvre des suites de chiffrement hybrides à échange de clés post-quantique AWS KMS qui combinent des éléments classiques et post-quantiques afin de garantir que votre connexion TLS est au moins aussi solide qu'elle le serait avec les suites de chiffrement classiques.

Ces suites de chiffrement hybrides peuvent être utilisées sur vos charges de travail de production dans la plupart des cas. Régions AWS Cependant, étant donné que les caractéristiques de performance et les exigences en bande passante des suites de chiffrement hybrides sont différentes de celles des mécanismes d'échange de clés classiques, nous vous recommandons de les tester sur vos appels d' AWS KMS API dans des conditions différentes.

Commentaires

Comme toujours, nous accueillons vos commentaires et votre participation à nos référentiels open-source. Nous aimerions en particulier savoir comment votre infrastructure interagit avec cette nouvelle variante du trafic TLS.

- Pour nous faire part de vos commentaires sur ce point, utilisez le lien Commentaires dans le coin supérieur droit de cette page.
- Nous développons ces suites de chiffrement hybrides en open source dans le s2n-tlsréférentiel sur GitHub. Pour fournir des commentaires sur l'utilisabilité des suites de chiffrement, ou pour partager de nouvelles conditions de test ou de nouveaux résultats, créez un problème dans s2n-tls repository.
- Nous écrivons des exemples de code pour utiliser le TLS post-quantique hybride avec AWS KMS dans le aws-kms-pq-tls-example GitHub référentiel. Pour poser des questions ou partager des

idées sur la configuration de votre client HTTP ou de votre AWS KMS client pour utiliser les suites de chiffrement hybrides, créez un problème dans aws-kms-pq-tls-example repository.

Soutenu Régions AWS

Le protocole TLS post-quantique pour AWS KMS est disponible sur tous les Régions AWS AWS KMS supports, à l'exception de la Chine (Pékin) et de la Chine (Ningxia).

> **Note**
>
> AWS KMS ne prend pas en charge le protocole TLS post-quantique hybride pour les points de terminaison FIPS dans. AWS GovCloud (US)

Pour obtenir la liste des AWS KMS points de terminaison de chacun Région AWS, consultez la section AWS Key Management Service Points de terminaison et quotas dans le. Référence générale d'Amazon Web Services Pour plus d'informations sur les points de terminaison FIPS, consultez Points de terminaison FIPS dans le Référence générale d'Amazon Web Services.

À propos de l'échange de clés post-quantiques hybrides dans TLS

AWS KMS prend en charge les suites de chiffrement hybrides à échange de clés post-quantique. Vous pouvez utiliser le AWS Common Runtime AWS SDK for Java 2.x et le Common Runtime sur les systèmes Linux pour configurer un client HTTP qui utilise ces suites de chiffrement. Ensuite, chaque fois que vous vous connectez à un AWS KMS point de terminaison avec votre client HTTP, les suites de chiffrement hybrides sont utilisées.

Ce client HTTP utilise s2n-tls, qui est une implémentation open source du protocole TLS. Les suites de chiffrement hybrides qui s2n-tls les utilisations sont mises en œuvre uniquement pour l'échange de clés, et non pour le chiffrement direct des données. Pendant l'échange de clés, le client et le serveur calculent la clé qu'ils utiliseront pour chiffrer et déchiffrer les données sur le réseau.

Les algorithmes qui s2n-tls les utilisations sont un hybride qui combine Elliptic Curve Diffie-Hellman (ECDH), un algorithme d'échange de clés classique utilisé aujourd'hui dans le TLS, avec un mécanisme d'encapsulation de clés basé sur un réseau de modules (ML-KEM), un algorithme de chiffrement à clé publique et d'établissement de clés que le National Institute for Standards and Technology (NIST) a désigné comme son premier algorithme standard d'accord de clé post-quantique. Ce mécanisme hybride utilise chacun des algorithmes indépendamment pour générer une clé. Ensuite, il combine les deux clés cryptographiquement. Avec s2n-tls, vous pouvez configurer un

client HTTP pour préférer le protocole TLS post-quantique, qui place l'ECDH avec ML-KEM premier dans la liste des préférences. Les algorithmes d'échange de clés classiques sont inclus dans la liste des préférences pour garantir la compatibilité, mais ils sont plus bas dans l'ordre de préférence.

Utilisation du TLS post-quantique hybride avec AWS KMS

Vous pouvez utiliser le protocole TLS post-quantique hybride pour vos appels vers. AWS KMS Lors de la configuration de votre environnement de test client HTTP, tenez compte des informations suivantes :

Chiffrement en transit

Les suites de chiffrement hybrides de s2n-tls ne sont utilisés que pour le chiffrement en transit. Ils protègent vos données pendant leur transfert entre votre client et le AWS KMS terminal. AWS KMS n'utilise pas ces suites de chiffrement pour chiffrer les données sous. AWS KMS keys

Au lieu de cela, lorsque vous AWS KMS cryptez vos données sous des clés KMS, il utilise une cryptographie symétrique avec des clés de 256 bits et l'algorithme AES-GCM (Advanced Encryption Standard in Galois Counter Mode), qui est déjà résistant aux attaques quantiques. Dans un avenir théorique, les attaques de calcul quantique à grande échelle sur les textes chiffrés créés sous les clés AES-GCM 256 bits réduiront la sécurité effective de la clé à 128 bits. Ce niveau de sécurité est suffisant pour rendre irréalisables les attaques par force brute sur des AWS KMS textes chiffrés.

Systèmes pris en charge

Utilisation des suites de chiffrement hybrides dans s2n-tls n'est actuellement pris en charge que sur les systèmes Linux. En outre, ces suites de chiffrement ne sont prises en charge SDKs que dans la mesure où elles prennent en charge le AWS Common Runtime, comme le AWS SDK for Java 2.x. Pour obtenir un exemple, consultez Configurer le TLS post-quantique hybride.

AWS KMS Points de terminaison

Lorsque vous utilisez les suites de chiffrement hybrides, utilisez le point de AWS KMS terminaison standard. AWS KMS ne prend pas en charge le protocole TLS post-quantique hybride pour les points de terminaison validés par la norme FIPS 140-3.

Lorsque vous configurez un client HTTP pour préférer les connexions TLS post-quantiques avec s2n-tls, les chiffrements post-quantiques sont les premiers de la liste des préférences chiffrées. Toutefois, la liste des préférences inclut les chiffrements classiques non hybrides plus bas dans l'ordre de préférence pour la compatibilité. Lorsque vous configurez un client HTTP pour préférer le protocole

TLS post-quantique avec un point de terminaison validé AWS KMS FIPS 140-3, s2n-tls négocie un code d'échange de clés classique et non hybride.

Pour obtenir la liste des AWS KMS points de terminaison de chacun Région AWS, consultez la section AWS Key Management Service Points de terminaison et quotas dans le. Référence générale d'Amazon Web Services Pour plus d'informations sur les points de terminaison FIPS, consultez Points de terminaison FIPS dans le Référence générale d'Amazon Web Services.

Performances attendues

Nos premiers tests de référence montrent que les suites de chiffrement hybrides dans s2n-tls sont plus lents que les suites de chiffrement TLS classiques. L'effet varie en fonction du profil réseau, de la vitesse du processeur, du nombre de cœurs et de votre fréquence d'appel. Pour plus d'informations, consultez le plan de migration de la cryptographie AWS post-quantique.

Configurer le TLS post-quantique hybride

Dans cette procédure, ajoutez une dépendance Maven pour le client HTTP AWS Common Runtime. Vous pouvez ensuite configurer un client HTTP qui privilégie le protocole TLS post-quantique. Créez ensuite un AWS KMS client qui utilise le client HTTP.

Pour voir des exemples pratiques complets de configuration et d'utilisation du TLS post-quantique hybride avec AWS KMS, consultez le aws-kms-pq-tls-exampleréférentiel.

> **Note**
>
> Le client HTTP AWS Common Runtime, disponible en version préliminaire, est devenu généralement disponible en février 2023. Dans cette version, la classe `tlsCipherPreference` et le paramètre de méthode `tlsCipherPreference()` sont remplacés par le paramètre de méthode `postQuantumTlsEnabled()`. Si vous utilisiez cet exemple dans la version préliminaire, vous devez procéder à la mise à jour de votre code.

1. Ajoutez le client AWS Common Runtime à vos dépendances Maven. Nous vous recommandons d'utiliser la dernière version disponible.

   Par exemple, cette instruction ajoute une version `2.30.22` du client AWS Common Runtime à vos dépendances Maven.

   ```
   <dependency>
   ```

```
    <groupId>software.amazon.awssdk</groupId>
    <artifactId>aws-crt-client</artifactId>
    <version>2.30.22</version>
</dependency>
```

2. Pour activer les suites de chiffrement post-quantiques hybrides, ajoutez-les AWS SDK for Java 2.x à votre projet et initialisez-le. Activez ensuite les suites de chiffrement post-quantique hybrides sur votre client HTTP, comme indiqué dans l'exemple suivant.

   Ce code utilise le paramètre `postQuantumTlsEnabled()` method pour configurer un client HTTP d'exécution AWS commun qui préfère la suite de chiffrement post-quantique hybride recommandée, ECDH avec ML-KEM. Il utilise ensuite le client HTTP configuré pour créer une instance du client AWS KMS asynchrone. `KmsAsyncClient` Une fois ce code terminé, toutes les demandes d'API AWS KMS effectuées sur l'instance `KmsAsyncClient` utilisent le protocole TLS post-quantique hybride.

   ```
   // Configure HTTP client
   SdkAsyncHttpClient awsCrtHttpClient = AwsCrtAsyncHttpClient.builder()
              .postQuantumTlsEnabled(true)
              .build();

   // Create the AWS KMS async client
   KmsAsyncClient kmsAsync = KmsAsyncClient.builder()
             .httpClient(awsCrtHttpClient)
             .build();
   ```

3. Testez vos AWS KMS appels avec le protocole TLS post-quantique hybride.

   Lorsque vous appelez des opérations AWS KMS d'API sur le AWS KMS client configuré, vos appels sont transmis au point de AWS KMS terminaison à l'aide du protocole TLS post-quantique hybride. Pour tester votre configuration, appelez une AWS KMS API, telle que`ListKeys`.

   ```
   ListKeysReponse keys = kmsAsync.listKeys().get();
   ```

Testez votre configuration TLS post-quantique hybride

Envisagez d'exécuter les tests suivants avec des suites de chiffrement hybrides sur les applications qui appellent AWS KMS.

- Exécutez des tests de charge et des repères. Les suites de chiffrement hybrides fonctionnent différemment des algorithmes d'échange de clés traditionnels. Il se peut que vous deviez ajuster les délais d'expiration de votre connexion pour tenir compte des durées de liaison plus longues. Si vous exécutez une AWS Lambda fonction, prolongez le paramètre de délai d'exécution.
- Essayez de vous connecter à partir de différents endroits. Selon le chemin réseau emprunté par votre demande, vous découvrirez peut-être que des hôtes intermédiaires, des proxys ou des pare-feu avec inspection approfondie des paquets (DPI) bloquent la demande. Cela peut être dû à l'utilisation des nouvelles suites de chiffrement dans le ClientHellocadre de la poignée de main TLS ou à des messages d'échange de clés plus volumineux. Si vous rencontrez des difficultés pour résoudre ces problèmes, collaborez avec votre équipe de sécurité ou les administrateurs informatiques pour mettre à jour la configuration appropriée et débloquer les nouvelles suites de chiffrement TLS.

En savoir plus sur le TLS post-quantique dans AWS KMS

Pour plus d'informations sur l'utilisation du protocole TLS post-quantique hybride dans AWS KMS, consultez les ressources suivantes.

- Pour en savoir plus sur la cryptographie post-quantique sur AWS, y compris des liens vers des articles de blog et des articles de recherche, voir Cryptographie post-quantique.
- Pour plus d'informations sur s2n-tls, voir Présentation s2n-tls, une nouvelle implémentation du protocole TLS open source et utilisation s2n-tls.
- Pour plus d'informations sur le client HTTP AWS Common Runtime, consultez la section Configuration du client HTTP AWS CRT dans le guide du AWS SDK for Java 2.x développeur.
- Pour de plus amples informations sur le projet de chiffrement post-quantique du National Institute for Standards and Technology (NIST), veuillez consulter Chiffrement post-quantique.
- Pour plus d'informations sur la normalisation du chiffrement post-quantique par le NIST, consultez la page Post-Quantum Cryptography Standardization (Normalisation du chiffrement post-quantique).

Connectez-vous AWS KMS via un point de terminaison VPC

Vous pouvez vous connecter directement AWS KMS via un point de terminaison d'interface privée dans votre cloud privé virtuel (VPC). Lorsque vous utilisez un point de terminaison VPC d'interface, la communication entre votre VPC et celui-ci AWS KMS s'effectue entièrement au sein du réseau. AWS

AWS KMS prend en charge les points de terminaison Amazon Virtual Private Cloud (Amazon VPC) alimentés par. AWS PrivateLink Chaque point de terminaison VPC est représenté par une ou plusieurs interfaces réseau élastiques (ENIs) avec des adresses IP privées dans vos sous-réseaux VPC.

Le point de terminaison VPC de l'interface connecte directement votre VPC AWS KMS sans passerelle Internet, périphérique NAT, connexion VPN ou connexion. AWS Direct Connect Les instances de votre VPC n'ont pas besoin d'adresses IP publiques pour communiquer avec elles. AWS KMS

Régions

AWS KMS prend en charge les points de terminaison VPC et les politiques de points de terminaison VPC dans tous Régions AWS ceux qui sont pris en charge. AWS KMS

Considérations relatives aux points de AWS KMS terminaison VPC

Avant de configurer un point de terminaison VPC d'interface pour AWS KMS, consultez la rubrique Propriétés et limites du point de terminaison d'interface dans le AWS PrivateLink Guide.

AWS KMS la prise en charge d'un point de terminaison VPC inclut les éléments suivants.

- Vous pouvez utiliser votre point de terminaison d'un VPC pour appeler toutes les opérations d'API AWS KMS à partir de votre VPC.
- Vous pouvez créer un point de terminaison VPC d'interface qui se connecte à un point de terminaison AWS KMS régional ou à un point de terminaison AWS KMS FIPS.
- Vous pouvez utiliser AWS CloudTrail les journaux pour vérifier votre utilisation des clés KMS via le point de terminaison VPC. Pour plus de détails, consultez Journalisation AWS KMS des demandes utilisant un point de terminaison VPC.

Rubriques

- Créez un point de terminaison VPC pour AWS KMS
- Connectez-vous à un point de AWS KMS terminaison VPC
- Utiliser les points de terminaison VPC pour contrôler l'accès aux ressources AWS KMS
- Journalisation AWS KMS des demandes utilisant un point de terminaison VPC

Créez un point de terminaison VPC pour AWS KMS

Vous pouvez créer un point de terminaison VPC pour AWS KMS en utilisant la console Amazon VPC ou l'API Amazon VPC. Suivez les procédures pour créer un point de terminaison d'interface à l'aide de l'une des valeurs suivantes.

- Pour créer un point de terminaison VPC pour AWS KMS, utilisez le nom de service suivant :

  ```
  com.amazonaws.region.kms
  ```

  Par exemple, dans la région USA Ouest (Oregon) (`us-west-2`), le nom du service serait :

  ```
  com.amazonaws.us-west-2.kms
  ```

- Pour créer un point de terminaison d'un VPC dans connecté à un point de terminaison FIPS AWS KMS, utilisez le nom de service suivant :

  ```
  com.amazonaws.region.kms-fips
  ```

  Par exemple, dans la région USA Ouest (Oregon) (`us-west-2`), le nom du service serait :

  ```
  com.amazonaws.us-west-2.kms-fips
  ```

Pour faciliter l'utilisation du point de terminaison de VPC, vous pouvez activer un nom DNS privé pour votre point de terminaison d'un VPC. Si vous sélectionnez l'option Enable DNS Name (Activer le nom DNS), le nom d'hôte DNS AWS KMS standard est résolu vers votre point de terminaison d'un VPC. Par exemple, `https://kms.us-west-2.amazonaws.com` serait résolu vers un point de terminaison d'un VPC connecté au nom du service `com.amazonaws.us-west-2.kms`.

Cette option facilite l'utilisation du point de terminaison d'un VPC. Les AWS SDKs et AWS CLI utilisent le nom d'hôte AWS KMS DNS standard par défaut. Vous n'avez donc pas besoin de spécifier l'URL du point de terminaison du VPC dans les applications et les commandes.

Pour de plus amples informations, veuillez consulter Accès à un service via un point de terminaison d'interface dans le Guide AWS PrivateLink .

Connectez-vous à un point de AWS KMS terminaison VPC

Vous pouvez vous connecter AWS KMS via le point de terminaison VPC à l'aide d'un AWS SDK, du AWS CLI ou. Outils AWS pour PowerShell Pour spécifier le point de terminaison VPC, utilisez son nom DNS.

Par exemple, cette commande list-keys utilise le paramètre `endpoint-url` pour indiquer le point de terminaison VPC. Pour utiliser une commande comme celle-ci, remplacez l'exemple d'ID de point de terminaison VPC par celui de votre compte.

```
$ aws kms list-keys --endpoint-url https://vpce-1234abcdf5678c90a-09p7654s-us-east-1a.ec2.us-east-1.vpce.amazonaws.com
```

Autorisations requises

Pour qu'une AWS KMS demande utilisant un point de terminaison VPC aboutisse, le principal a besoin d'autorisations provenant de deux sources :

- Une politique de clé, une politique IAM ou un octroi doivent accorder au principal l'autorisation d'appeler l'opération sur la ressource (clé KMS ou alias).
- Une politique de point de terminaison d'un VPC doit accorder au principal l'autorisation d'utiliser le point de terminaison pour effectuer la demande.

Par exemple, une politique de clé peut accorder à un principal l'autorisation d'appeler Decrypt sur une clé KMS particulière. Toutefois, la politique de point de terminaison d'un VPC peut ne pas permettre à ce principal d'appeler `Decrypt` sur cette clé KMS à l'aide du point de terminaison.

Une politique de point de terminaison VPC peut également autoriser un principal à utiliser le point de terminaison pour appeler certaines DisableKeyclés KMS. Mais si le principal ne dispose pas de ces autorisations provenant d'une politique de clé, d'une politique IAM ou d'un octroi, la demande échoue.

Vous pouvez créer une politique de point de terminaison VPC lorsque vous créez votre point de terminaison, et vous pouvez modifier la politique de point de terminaison d'un VPC à tout moment. Utilisez la console de gestion VPC ou les opérations CreateVpcEndpointor ModifyVpcEndpoint. Vous pouvez également créer et modifier une politique de point de terminaison VPC à l'aide d'un AWS CloudFormation modèle. Pour obtenir de l'aide sur l'utilisation de la console de gestion de VPC, veuillez consulter Créer un point de terminaison d'interface et Modification d'un point de terminaison d'interface dans le AWS PrivateLink Guide .

Noms d'hôtes privés

Si vous avez activé les noms d'hôte privés lorsque vous avez créé votre point de terminaison VPC, vous n'avez pas besoin de spécifier l'URL de point de terminaison VPC dans vos commandes de CLI ou dans la configuration de l'application. Le nom d'hôte DNS AWS KMS standard est résolu vers votre point de terminaison d'un VPC. Le AWS CLI et SDKs utilisez ce nom d'hôte par défaut, afin que vous puissiez commencer à utiliser le point de terminaison VPC pour vous connecter à AWS KMS un point de terminaison régional sans rien modifier dans vos scripts et applications.

Pour utiliser des noms d'hôte privés, les attributs `enableDnsHostnames` et `enableDnsSupport` de votre VPC doivent avoir la valeur `true`. Pour définir ces attributs, utilisez l'ModifyVpcAttributeopération. Pour plus d'informations, veuillez consulter Afficher et mettre à jour les attributs DNS pour votre VPC dans le Guide de l'utilisateur Amazon VPC.

Utiliser les points de terminaison VPC pour contrôler l'accès aux ressources AWS KMS

Vous pouvez contrôler l'accès aux AWS KMS ressources et aux opérations lorsque la demande provient d'un VPC ou utilise un point de terminaison VPC. Pour ce faire, utilisez l'une des clés de condition globales suivantes dans une politique de clé ou une politique IAM.

- Utilisez la clé de condition `aws:sourceVpce` pour accorder ou restreindre l'accès en fonction du point de terminaison d'un VPC.
- Utilisez la clé de condition `aws:sourceVpc` pour accorder ou restreindre l'accès en fonction du VPC qui héberge le point de terminaison privé.

> **Note**
>
> Soyez prudent lorsque vous créez des politiques de clé et des politiques IAM basées sur votre point de terminaison d'un VPC. Si une déclaration de politique exige que les demandes proviennent d'un VPC ou d'un point de terminaison VPC en particulier, les demandes provenant de AWS services intégrés qui utilisent une AWS KMS ressource en votre nom risquent d'échouer. Pour obtenir de l'aide, veuillez consulter Utilisation de conditions de point de terminaison d'un VPC dans des politiques avec des autorisations AWS KMS.
> Par ailleurs, la clé de condition `aws:sourceIP` n'est pas en vigueur lorsque la demande provient d'un point de terminaison d'un VPC Amazon. Pour restreindre les requêtes à un point

de terminaison VPC, utilisez les clés de condition `aws:sourceVpce` ou `aws:sourceVpc`. Pour de plus amples informations, veuillez consulter Gestion des identités et des accès pour les points de terminaison d'un VPC et les services de points de terminaison d'un VPC dans le Guide AWS PrivateLink .

Vous pouvez utiliser ces clés de condition globales pour contrôler l'accès aux AWS KMS keys (clés KMS), aux alias et aux opérations de CreateKeyce type qui ne dépendent d'aucune ressource en particulier.

Par exemple, l'exemple de politique de clé suivant autorise un utilisateur à effectuer des opérations de chiffrement à l'aide d'une clé KMS uniquement lorsque la demande provient du point de terminaison d'un VPC spécifié. Lorsqu'un utilisateur fait une demande à AWS KMS, l'ID du point de terminaison VPC figurant dans la demande est comparé à la valeur de la clé de `aws:sourceVpce` condition dans la politique. S'il n'y a pas de concordance, la requête est refusée.

Pour utiliser une politique comme celle-ci, remplacez l' Compte AWS identifiant réservé et le point de IDs terminaison VPC par des valeurs valides pour votre compte.

```
{
    "Id": "example-key-1",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Enable IAM policies",
            "Effect": "Allow",
            "Principal": {"AWS":["111122223333"]},
            "Action": ["kms:*"],
            "Resource": "*"
        },
        {
            "Sid": "Restrict usage to my VPC endpoint",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*"
            ],
            "Resource": "*",
```

```
            "Condition": {
                "StringNotEquals": {
                    "aws:sourceVpce": "vpce-1234abcdf5678c90a"
                }
            }
        }

    ]
}
```

Vous pouvez également utiliser la clé de condition `aws:sourceVpc` pour restreindre l'accès à vos clés KMS en fonction du VPC dans lequel réside le point de terminaison d'un VPC.

L'exemple de politique de clé suivant autorise des commandes qui gèrent la clé KMS uniquement lorsqu'ils proviennent de `vpc-12345678`. En outre, il autorise les commandes qui utilisent la clé KMS pour des opérations cryptographiques uniquement lorsqu'elles proviennent de `vpc-2b2b2b2b`. Vous pouvez utiliser politique comme celle-ci si une application est en cours d'exécution dans un VPC, mais que vous utilisez un second VPC isolé pour les fonctions de gestion.

Pour utiliser une politique comme celle-ci, remplacez l' Compte AWS identifiant réservé et le point de IDs terminaison VPC par des valeurs valides pour votre compte.

```
{
    "Id": "example-key-2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow administrative actions from vpc-12345678",
            "Effect": "Allow",
            "Principal": {"AWS": "111122223333"},
            "Action": [
                "kms:Create*","kms:Enable*","kms:Put*","kms:Update*",
                "kms:Revoke*","kms:Disable*","kms:Delete*",
                "kms:TagResource", "kms:UntagResource"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-12345678"
                }
            }
        },
```

```
        {
            "Sid": "Allow key usage from vpc-2b2b2b2b",
            "Effect": "Allow",
            "Principal": {"AWS": "111122223333"},
            "Action": [
                "kms:Encrypt","kms:Decrypt","kms:GenerateDataKey*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-2b2b2b2b"
                }
            }
        },
        {
            "Sid": "Allow read actions from everywhere",
            "Effect": "Allow",
            "Principal": {"AWS": "111122223333"},
            "Action": [
                "kms:Describe*","kms:List*","kms:Get*"
            ],
            "Resource": "*",
        }
    ]
}
```

Journalisation AWS KMS des demandes utilisant un point de terminaison VPC

AWS CloudTrail enregistre toutes les opérations qui utilisent le point de terminaison VPC. Lorsqu'une demande AWS KMS utilise un point de terminaison VPC, l'ID du point de terminaison du VPC apparaît dans l'entrée du AWS CloudTrail journal qui enregistre la demande. Vous pouvez utiliser l'identifiant du point de terminaison pour auditer l'utilisation de votre point de terminaison AWS KMS VPC.

Toutefois, vos CloudTrail journaux n'incluent pas les opérations demandées par les principaux sur d'autres comptes ni les demandes d' AWS KMS opérations sur les clés KMS et les alias sur d'autres comptes. De plus, pour protéger votre VPC, les demandes refusées par une politique de point de terminaison du VPC, mais qui auraient autrement été autorisées, ne sont pas enregistrées dans. AWS CloudTrail

Par exemple, cet exemple d'entrée de journal enregistre une requête GenerateDataKey qui utilise le point de terminaison d'un VPC. Le champ `vpcEndpointId` apparaît à la fin de l'entrée de journal.


```
{
  "eventVersion":"1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "accountId": "111122223333",
    "userName": "Alice"
  },
  "eventTime":"2018-01-16T05:46:57Z",
  "eventSource":"kms.amazonaws.com",
  "eventName":"GenerateDataKey",
  "awsRegion":"eu-west-1",
  "sourceIPAddress":"172.01.01.001",
  "userAgent":"aws-cli/1.14.23 Python/2.7.12 Linux/4.9.75-25.55.amzn1.x86_64
 botocore/1.8.27",
  "requestParameters":{
    "keyId":"1234abcd-12ab-34cd-56ef-1234567890ab",
    "numberOfBytes":128
  },
  "responseElements":null,
  "requestID":"a9fff0bf-fa80-11e7-a13c-afcabff2f04c",
  "eventID":"77274901-88bc-4e3f-9bb6-acf1c16f6a7c",
  "readOnly":true,
  "resources":[{
    "ARN":"arn:aws:kms:eu-
west-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId":"111122223333",
    "type":"AWS::KMS::Key"
  }],
  "eventType":"AwsApiCall",
  "recipientAccountId":"111122223333",
  "vpcEndpointId": "vpce-1234abcdf5678c90a"
}
```

Support des terminaux à double pile

AWS KMS fournit un point de terminaison public à double pile qui prend en charge à la fois les IPv6 clients IPv4 et les clients. Un point de terminaison à double pile permet aux clients de communiquer avec eux AWS KMS en utilisant l'une IPv4 ou l'autre IPv6 des adresses. Pour plus d'informations sur les points de AWS KMS terminaison, consultez la section Points de AWS Key Management Service terminaison et quotas.

Le point de terminaison public à AWS KMS double pile `https://kms.`*`your-region`*`.api.aws` prend en charge à la fois IPv4 les IPv6 clients. AWS KMS est également accessible en privé via IPv4 et IPv6 depuis votre cloud privé virtuel (VPC) à l'aide de. AWS PrivateLink Pour plus d'informations sur la création de points de terminaison VPC d'interface privée pour AWS KMS, consultez. Connectez-vous AWS KMS via un point de terminaison VPC

Pour plus d'informations sur l' IPv6 adressage qui vous convient VPCs, consultez la section Comment fonctionne Amazon VPC dans le guide de l'utilisateur d'Amazon Virtual Private Cloud. Pour plus d'informations sur la configuration de votre VPC en mode double pile, consultez la section Adressage IP pour votre réseau VPCs et vos sous-réseaux dans le guide de l'utilisateur Amazon Virtual Private Cloud.

Fonctionnalités non disponibles sur IPv6

AWS KMS Impossible de communiquer IPv6 avec les magasins de AWS CloudHSM clés ou les magasins de clés externes. Cette limitation ne vous empêche pas d' AWS KMS APIs appeler IPv6.

AWS KMS concepts

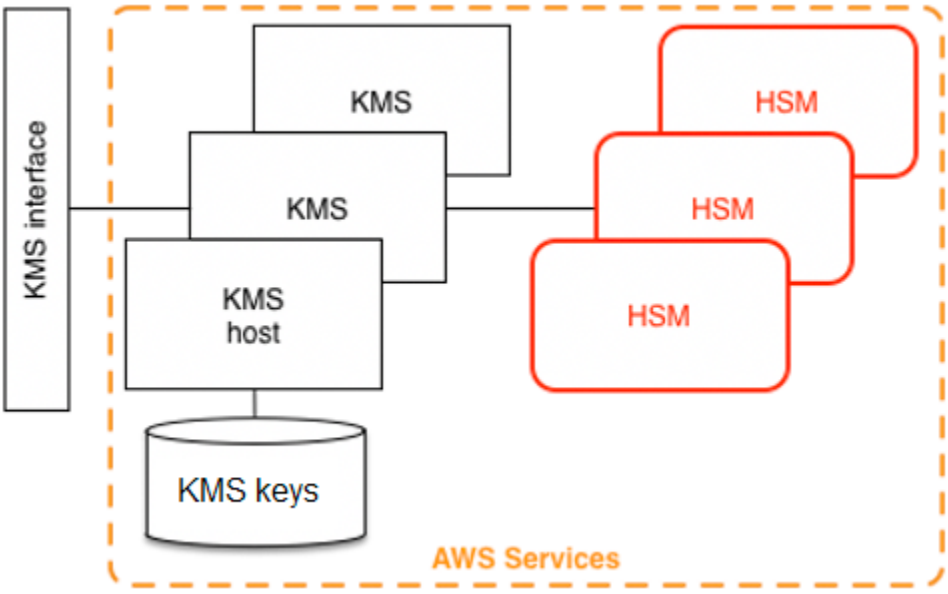
Découvrez les termes et concepts de base utilisés dans AWS Key Management Service (AWS KMS) et découvrez comment ils fonctionnent ensemble pour protéger vos données.

Présentation de AWS KMS

AWS Key Management Service (AWS KMS) fournit une interface Web pour générer et gérer des clés cryptographiques et fonctionne en tant que fournisseur de services cryptographiques pour la protection des données. AWS KMS propose des services traditionnels de gestion des clés intégrés à AWS des services visant à fournir une vue cohérente des clés des clients AWS, avec une gestion et un audit centralisés.

AWS KMS inclut une interface Web via l' AWS Management Console interface de ligne de commande et des opérations d' RESTfulAPI pour demander les opérations cryptographiques d'un parc distribué de modules de sécurité matériels validés par la norme FIPS 140-3 (). HSMs Le AWS KMS HSM est une appliance cryptographique matérielle autonome multipuce conçue pour fournir des fonctions cryptographiques dédiées répondant aux exigences de sécurité et d'évolutivité de. AWS KMS Vous pouvez établir votre propre hiérarchie cryptographique basée sur le HSM sous les clés que vous gérez en tant que AWS KMS keys. Ces clés ne sont disponibles que sur le HSMs et uniquement en mémoire pendant le temps nécessaire au traitement de votre demande cryptographique. Vous pouvez créer plusieurs clés KMS, chacune représentée par son ID de clé. Ce n'est que dans le cadre des rôles et des comptes AWS IAM administrés par chaque client que les clés KMS gérées par le client peuvent être créées, supprimées ou utilisées pour chiffrer, déchiffrer, signer ou vérifier des données. Vous pouvez définir des contrôles d'accès pour déterminer qui peut gérer and/or l'utilisation des clés KMS en créant une politique attachée à la clé. Ces politiques vous permettent de définir des utilisations propres à l'application de vos clés pour chaque opération d'API.

En outre, la plupart des AWS services prennent en charge le chiffrement des données au repos à l'aide de clés KMS. Cette fonctionnalité permet aux clients de contrôler comment et quand les AWS services peuvent accéder aux données chiffrées en contrôlant comment et quand les clés KMS sont accessibles.



AWS KMS est un service à plusieurs niveaux composé d' AWS KMS hôtes accessibles sur le Web et d'un niveau de. HSMs Le regroupement de ces hôtes hiérarchisés forme la AWS KMS pile. Toutes les demandes AWS KMS doivent être effectuées via le protocole TLS (Transport Layer Security) et se terminer sur un AWS KMS hôte. AWS KMS les hôtes n'autorisent le TLS qu'avec une suite de chiffrement qui assure une parfaite confidentialité des transmissions. AWS KMS authentifie et autorise vos demandes en utilisant les mêmes mécanismes d'identification et de politique AWS Identity and Access Management (IAM) que ceux disponibles pour toutes les autres opérations d'API. AWS

AWS KMS objectifs de conception

AWS KMS est conçu pour répondre aux exigences suivantes.

Durabilité

La durabilité des clés cryptographiques est conçue pour égaler celle des services les plus durables en AWS. Une seule clé cryptographique peut chiffrer de grands volumes de vos données qui se sont accumulés sur une longue période.

Digne de confiance

L'utilisation des clés est protégée par des politiques de contrôle d'accès que vous définissez et gérez. Il n'existe aucun mécanisme permettant d'exporter des clés KMS en texte brut. La confidentialité de vos clés cryptographiques est primordiale. Plusieurs employés d'Amazon

disposant d'un accès spécifique à un rôle aux contrôles d'accès basés sur le quorum sont tenus d'effectuer des actions administratives sur le. HSMs

Faible latence et haut débit

AWS KMS fournit des opérations cryptographiques à des niveaux de latence et de débit adaptés à une utilisation par d'autres services dans. AWS

Régions indépendantes

AWS fournit des régions indépendantes aux clients qui ont besoin de restreindre l'accès aux données dans différentes régions. L'utilisation de la clé peut être isolée dans un Région AWS.

Source sécurisée de nombres aléatoires

Parce qu'une cryptographie puissante dépend d'une génération de nombres aléatoires vraiment imprévisible, elle AWS KMS fournit une source validée de nombres aléatoires de haute qualité.

Audit

AWS KMS enregistre l'utilisation et la gestion des clés cryptographiques dans des AWS CloudTrail journaux. Vous pouvez utiliser AWS CloudTrail les journaux pour contrôler l'utilisation de vos clés cryptographiques, y compris l'utilisation des clés par les AWS services en votre nom.

Pour atteindre ces objectifs, le AWS KMS système inclut un ensemble d' AWS KMS opérateurs et d'opérateurs hôtes de services (collectivement, les « opérateurs ») qui administrent les « domaines ». Un domaine est un ensemble de AWS KMS serveurs et d'opérateurs défini au niveau régional. HSMs Chaque AWS KMS opérateur dispose d'un jeton matériel qui contient une paire de clés privée et publique utilisée pour authentifier ses actions. Ils HSMs disposent d'une paire de clés privée et publique supplémentaire pour établir des clés de chiffrement qui protègent la synchronisation de l'état du HSM.

AWS KMS keys

Les clés KMS que vous créez et gérez pour les utiliser dans vos propres applications cryptographiques sont du type connu sous le nom de clés gérées par le client. Les clés gérées par le client peuvent également être utilisées conjointement avec AWS des services qui utilisent des clés KMS pour chiffrer les données que le service stocke en votre nom. Les clés gérées par le client sont recommandées aux clients qui souhaitent avoir un contrôle total sur le cycle de vie et l'utilisation de leurs clés. L'ajout d'une clé gérée par le client à votre compte entraîne des frais mensuels. De

plus, les demandes d' and/or utilisation de la clé entraînent un coût d'utilisation. Consultez AWS Key Management Service les tarifs pour plus de détails.

Dans certains cas, un client peut souhaiter qu'un AWS service crypte ses données, mais il ne souhaite pas avoir à gérer les clés et ne veut pas payer pour une clé. Une Clé gérée par AWSest une clé KMS qui existe dans votre compte, mais qui ne peut être utilisée que dans certaines circonstances. Plus précisément, elle ne peut être utilisée que dans le contexte du AWS service dans lequel vous opérez et elle ne peut être utilisée que par les principaux titulaires du compte sur lequel la clé existe. Vous ne pouvez rien gérer concernant le cycle de vie ou les autorisations de ces clés. Lorsque vous utilisez les fonctionnalités de chiffrement dans les AWS services, vous pouvez le constater Clés gérées par AWS ; ils utilisent un alias de la forme « aws <service code>». Par exemple, une `aws/ebs` clé ne peut être utilisée que pour chiffrer les volumes EBS et uniquement pour les volumes utilisés par les principaux IAM sur le même compte que la clé. Pensez à un Clé gérée par AWS outil destiné uniquement aux utilisateurs de votre compte pour les ressources de votre compte. Vous ne pouvez pas partager de ressources chiffrées sous ou Clé gérée par AWS avec d'autres comptes. Bien que l'existence d'une clé Clé gérée par AWS soit gratuite dans votre compte, toute utilisation de ce type de clé vous est facturée par le AWS service attribué à la clé.

Clés gérées par AWS sont un ancien type de clé qui n'est plus créé pour les nouveaux AWS services à partir de 2021. Au lieu de cela, les nouveaux (et anciens) AWS services utilisent ce que l'on appelle un Clé détenue par AWSpour crypter les données des clients par défaut. An Clé détenue par AWS est une clé KMS qui se trouve dans un compte géré par le AWS service, de sorte que les opérateurs du service ont la possibilité de gérer son cycle de vie et ses autorisations d'utilisation. Grâce à l'utilisation Clés détenues par AWS, les AWS services peuvent crypter vos données de manière transparente et permettre un partage aisé des données entre comptes ou entre régions sans que vous ayez à vous soucier des autorisations clés. Clés détenues par AWS À utiliser pour les encryption-by-default charges de travail qui offrent une protection des données plus simple et plus automatisée. Étant donné que ces clés sont détenues et gérées par elles AWS, leur existence ou leur utilisation ne vous sont pas facturées, vous ne pouvez pas modifier leurs politiques, vous ne pouvez pas auditer les activités liées à ces clés et vous ne pouvez pas les supprimer. Utilisez des clés gérées par le client lorsque le contrôle est important, mais Clés détenues par AWS utilisez-les lorsque la commodité est primordiale.

	Clés gérées par le client	Clés gérées par AWS	Clés détenues par AWS

Stratégie de clé	Contrôlé exclusive ment par le client	Contrôlé par le service ; visible par le client	Contrôlé exclusive ment et consultab le uniquement par le AWS service qui crypte vos données
Journalisation	CloudTrail suivi des clients ou magasin de données sur les événements	CloudTrail suivi des clients ou magasin de données sur les événements	Non visible par le client
Gestion du cycle de vie	Le client gère la rotation, la suppressi on et l'emplacement régional	AWS KMS gère la rotation (annuelle), la suppression et la localisation régionale	Service AWS gère la rotation, la suppressi on et la localisation régionale
Tarification	Frais mensuels pour l'existence des clés (calculés au prorata de l'heure). Également facturé pour l'utilisation des clés	Aucuns frais mensuels, mais l'utilisation de l'API sur ces clés est facturée à l'appelant	Aucuns frais pour le client

Les clés KMS que vous créez sont des clés gérées par le client. Les Services AWS qui utilisent des clés KMS pour chiffrer vos ressources de service créent généralement des clés automatiquement. Les clés KMS Services AWS créées dans votre AWS compte sont Clés gérées par AWS. Les clés KMS Services AWS créées dans un compte de service sont Clés détenues par AWS.

Type de clé KMS	Peut afficher les métadonnées de clés KMS	Peut gérer une clé KMS	Utilisé uniquemen t pour mon Compte AWS	Rotation automatique	Tarification
Clé gérée par le client	Oui	Oui	Oui	Facultatif.	Frais mensuels

Type de clé KMS	Peut afficher les métadonnées de clés KMS	Peut gérer une clé KMS	Utilisé uniquement pour mon Compte AWS	Rotation automatique	Tarification
					(au prorata horaire) Frais par utilisation
Clé gérée par AWS	Oui	Non	Oui	Obligatoire. Chaque année (environ 365 jours).	Aucun frais mensuel Frais par utilisation (certains Services AWS payent ces frais pour vous)
Clé détenue par AWS	Non	Non	Non	Service AWS Gère la stratégie de rotation.	Pas de frais

AWS les services qui s'intègrent AWS KMS diffèrent dans leur prise en charge des clés KMS. Certains AWS services cryptent vos données par défaut à l'aide d'un Clé détenue par AWS ou d'un Clé gérée par AWS. Certains AWS services prennent en charge les clés gérées par le client. D'autres AWS services prennent en charge tous les types de clés KMS pour vous permettre de disposer facilement d'une Clé détenue par AWS clé gérée par le client Clé gérée par AWS, de la visibilité ou du contrôle d'une clé gérée par le client. Pour obtenir des informations détaillées sur les options de chiffrement proposées par un AWS service, consultez la rubrique Chiffrement au repos du guide de l'utilisateur ou du guide du développeur du service.

Clés gérées par le client

Les clés KMS que vous créez sont des clés gérées par le client. Les clés gérées par le client sont des clés KMS Compte AWS que vous créez, détenez et gérez. Vous disposez d'un contrôle total sur ces clés KMS, y compris établir et maintenir leurs politiques de clé, les politiques IAM et les octrois, leur activation et leur désactivation, la rotation de leurs éléments de chiffrement, l'ajout de balises, la création d'alias qui font référence aux clés KMS, et la planification des clés KMS en vue de leur suppression.

Les clés gérées par le client apparaissent sur la page Clés gérées par le client de la AWS Management Console pour AWS KMS. Pour identifier définitivement une clé gérée par le client, utilisez l'DescribeKeyopération. Pour les clés gérées par le client, la valeur du champ `KeyManager` de la réponse `DescribeKey` est `CUSTOMER`.

Vous pouvez utiliser vos clés gérées par le client dans les opérations de chiffrement et auditer leur utilisation dans les journaux AWS CloudTrail . En outre, de nombreux services AWS qui s'intègrent à AWS KMS vous permettent de spécifier une clé gérée par le client pour protéger les données qu'ils stockent et gèrent pour vous.

Les clés gérées par le client entraînent des frais mensuels et des frais pour une utilisation au-delà de l'offre gratuite. Ils sont comptabilisés dans les AWS KMS quotas de votre compte. Pour plus d'informations, consultez Tarification AWS Key Management Service et Quotas.

Clés gérées par AWS

Clés gérées par AWSsont des clés KMS de votre compte créées, gérées et utilisées en votre nom par un AWS service intégré à AWS KMS.

Certains AWS services vous permettent de choisir une clé Clé gérée par AWS ou une clé gérée par le client pour protéger vos ressources dans le cadre de ce service. En général, à moins que vous ne soyez obligé de contrôler la clé de chiffrement qui protège vos ressources, une Clé gérée par AWS est un bon choix. Vous n'êtes pas obligé de créer ou de gérer la clé ou sa stratégie de clé, et il n'y a jamais de frais mensuel pour une Clé gérée par AWS.

Vous êtes autorisé à les consulter Clés gérées par AWS dans votre compte, à consulter leurs politiques clés et à vérifier leur utilisation dans AWS CloudTrail les journaux. Cependant, vous ne pouvez pas modifier leurs propriétés Clés gérées par AWS, les faire pivoter, modifier leurs politiques clés ou planifier leur suppression. De plus, vous ne pouvez pas Clés gérées par AWS les utiliser directement dans des opérations cryptographiques ; le service qui les crée les utilise en votre nom.

Les politiques de contrôle des ressources de votre organisation ne s'appliquent pas à Clés gérées par AWS.

Clés gérées par AWS apparaissent sur la Clés gérées par AWSpage du AWS Management Console formulaire AWS KMS. Vous pouvez également les identifier Clés gérées par AWS par leurs alias, dont le format `aws/service-name` est tel que`aws/redshift`. Pour identifier définitivement un Clés gérées par AWS, utilisez l'DescribeKeyopération. Pour les Clés gérées par AWS, la valeur du champ `KeyManager` de la réponse `DescribeKey` est `AWS`.

Tous Clés gérées par AWS font l'objet d'une rotation automatique chaque année. Vous ne pouvez pas modifier cette programmation de rotation.

> **Note**
>
> En mai 2022, le calendrier de rotation AWS KMS a été modifié, Clés gérées par AWS passant de tous les trois ans (environ 1 095 jours) à chaque année (environ 365 jours).

Il n'y a pas de frais mensuels pour Clés gérées par AWS. Ils peuvent être soumis à des frais d'utilisation au-delà du niveau gratuit, mais certains AWS services couvrent ces coûts pour vous. Pour plus de détails, reportez-vous à la rubrique Chiffrement au repos dans le Guide de l'utilisateur ou le guide du développeur du service. Pour plus d’informations, consultez Tarification AWS Key Management Service.

Clés gérées par AWS ne comptez pas dans les quotas de ressources le nombre de clés KMS dans chaque région de votre compte. Mais lorsqu'elles sont utilisées pour le compte d'un principal dans votre compte, ces clés KMS sont prises en compte dans les quotas de demandes. Pour en savoir plus, consultez Quotas.

Clés détenues par AWS

Clés détenues par AWSsont un ensemble de clés KMS qu'un AWS service possède et gère pour une utilisation multiple Comptes AWS. Bien qu' Clés détenues par AWS ils ne soient pas dans votre compte Compte AWS, tout AWS service peut utiliser un Clé détenue par AWS pour protéger les ressources de votre compte.

Certains AWS services vous permettent de choisir une clé Clé détenue par AWS ou une clé gérée par le client. En général, à moins que vous ne soyez obligé d'auditer ou de contrôler la clé de

chiffrement qui protège vos ressources, une Clé détenue par AWS est un bon choix. Clés détenues par AWS sont totalement gratuits (pas de frais mensuels ni de frais d'utilisation), ils ne sont pas pris en compte dans les AWS KMS quotas de votre compte et ils sont faciles à utiliser. Vous n'avez pas besoin de créer ou de maintenir la clé ou sa politique de clé.

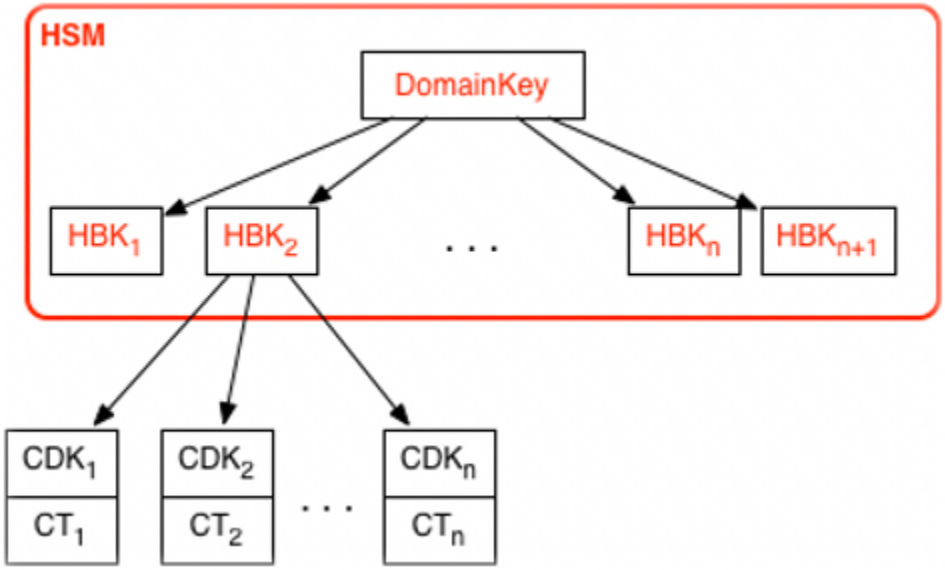
La rotation des services Clés détenues par AWS varie selon les services. Pour plus d'informations sur la rotation d'un service en particulier Clé détenue par AWS, consultez la rubrique Chiffrement au repos du guide de l'utilisateur ou du guide du développeur du service.

AWS KMS key hiérarchie

Votre hiérarchie de clés commence par une clé logique de haut niveau, une AWS KMS key. Une clé KMS représente un conteneur pour le matériel de clé de niveau supérieur et est définie de manière unique dans l'espace de noms service AWS avec un ARN (Amazon Resource Name). L'ARN comprend un identifiant de clé généré de manière unique, un ID de la clé. Une clé KMS est créée sur la base d'une demande initiée par l'utilisateur via AWS KMS. À réception, AWS KMS demande la création d'une clé de sauvegarde HSM initiale (HBK) à placer dans le conteneur de clés KMS. La clé HBK est générée sur un HSM du domaine et conçue pour ne jamais être exporté depuis le HSM en texte brut. Au lieu de cela, la clé HBK est exportée chiffrée dans des clés de domaine gérées par HSM. Ces jetons exportés HBKs sont appelés jetons clés exportés (EKTs).

L'EKT est exporté vers un stockage hautement durable et à faible latence. Par exemple, supposons que vous receviez un ARN sur la clé KMS logique. Cela représente le haut d'une hiérarchie de clés, ou contexte cryptographique, pour vous. Vous pouvez créer plusieurs clés KMS dans votre compte et définir des politiques sur vos clés KMS comme pour toute autre ressource AWS nommée.

Dans la hiérarchie d'une clé KMS spécifique, la clé HBK peut être considérée comme une version de la clé KMS. Lorsque vous souhaitez faire pivoter la clé KMS AWS KMS, une nouvelle clé HBK est créée et associée à la clé KMS en tant que HBK active pour la clé KMS. HBKs Les plus anciens sont conservés et peuvent être utilisés pour déchiffrer et vérifier des données précédemment protégées. Mais seule la clé cryptographique active peut être utilisée pour protéger de nouvelles informations.



Vous pouvez demander AWS KMS à utiliser vos clés KMS pour protéger directement les informations ou demander des clés supplémentaires générées par HSM qui sont protégées par votre clé KMS. Ces clés sont appelées clés de données client, ou CDKs. CDKs peut être renvoyé chiffré sous forme de texte chiffré (CT), en texte brut ou les deux. Tous les objets chiffrés sous une clé KMS (qu'il s'agisse de données fournies par le client ou de clés générées par HSM) ne peuvent être déchiffrés que sur un HSM via un appel. AWS KMS

Le texte chiffré renvoyé, ou la charge utile déchiffrée, n'y est jamais stocké. AWS KMS Les informations vous sont retournées via votre connexion TLS à AWS KMS. Cela s'applique également aux appels effectués par AWS les services en votre nom.

La hiérarchie des clés et les propriétés de ces clés spécifiques s’affichent dans le tableau suivant.

Clé	Description	Cycle de vie
Clé de domaine	Une clé AES-GCM 256 bits uniquement dans la mémoire d'une clé HSM utilisée pour envelopper les versions des clés KMS, les clés de sauvegarde HSM.	Rotation tous les jours[1]
Clé de sauvegarde HSM	Une clé symétrique 256 bits ou une clé privée RSA ou courbe elliptique, utilisée pour protéger les données et les clés du client stockées et	Rotation tous les ans[2] (configur ation facultative)

Clé	Description	Cycle de vie
	chiffrées sous les clés de domaine. Une ou plusieurs clés de sauvegarde HSM comprennent la clé KMS, représentée par l'ID KeyID.	
Clé de chiffrement dérivée	Une clé AES-GCM 256 bits résidant uniquement dans la mémoire d'une clé HSM est utilisée pour chiffrer les données et les clés du client. Dérivée d'une clé HBK pour chaque chiffrement.	Utilisée une seule fois par chiffrement et régénérée au déchiffrement
Clé de données client	Clé symétrique ou asymétrique définie par l'utilisateur, exportée depuis une clé HSM en texte brut et en texte chiffré. Chiffrée sous une clé de sauvegarde HSM et renvoyée aux utilisateurs autorisés via le canal TLS.	Rotation et utilisation contrôlée par application

[1] AWS KMS peut de temps à autre assouplir la rotation des clés de domaine à une rotation hebdomadaire au maximum pour tenir compte des tâches d'administration et de configuration du domaine.

[2] Les valeurs par défaut Clés gérées par AWS créées et gérées en votre AWS KMS nom font l'objet d'une rotation annuelle automatique.

Identifiants clés () KeyId

Les identificateurs de clé servent de noms pour vos clés KMS. Ils vous aident à reconnaître vos clés KMS dans la console. Vous les utilisez pour indiquer les clés KMS que vous souhaitez utiliser dans les opérations d'API AWS KMS , les politiques de clés, les politiques IAM et les octrois. Les valeurs de l'identifiant de clé ne sont absolument pas liées au matériel clé associé à la clé KMS.

AWS KMS définit plusieurs identificateurs clés. Lorsque vous créez une clé KMS, elle AWS KMS génère un ARN de clé et un ID de clé, qui sont des propriétés de la clé KMS. Lorsque vous créez un

alias, il AWS KMS génère un ARN d'alias basé sur le nom d'alias que vous définissez. Vous pouvez consulter les identifiants de clé et d'alias dans AWS Management Console et dans l' AWS KMS API.

Dans la AWS KMS console, vous pouvez afficher et filtrer les clés KMS en fonction de leur ARN clé, de leur ID de clé ou de leur nom d'alias, et les trier par ID de clé et nom d'alias. Pour obtenir de l'aide sur la recherche des identificateurs clés dans la console, veuillez consulter the section called “Trouvez l'ID et l'ARN de la clé”.

Dans l' AWS KMS API, les paramètres que vous utilisez pour identifier une clé KMS sont nommés `KeyId` ou une variante, telle que `TargetKeyId` ou`DestinationKeyId`. Cependant, les valeurs de ces paramètres ne sont pas limitées à la clé IDs. Certains peuvent prendre n'importe quel identifiant de clé valide. Pour plus d'informations sur les valeurs de chaque paramètre, consultez la description du paramètre dans la référence de l' AWS Key Management Service API.

> **Note**
>
> Lorsque vous utilisez l' AWS KMS API, faites attention à l'identifiant de clé que vous utilisez. Différents APIs nécessitent des identifiants de clé différents. En général, utilisez l'identificateur de clé le plus complet et le plus pratique pour votre tâche.

AWS KMS prend en charge les identificateurs de clé suivants.

ARN de clé

L'ARN de clé est l'Amazon Resource Name (ARN) d'une clé KMS. Il s'agit d'un identifiant unique et entièrement qualifié pour la clé KMS. Un ARN de clé inclut Compte AWS la région et l'ID de clé. Pour obtenir de l'aide sur la recherche de l'ARN de clé d'une clé KMS, veuillez consulter the section called “Trouvez l'ID et l'ARN de la clé”.

Le format d'un ARN de clé est le suivant :

```
arn:<partition>:kms:<region>:<account-id>:key/<key-id>
```

Voici un exemple d'ARN de clé pour une clé KMS de région unique.

```
arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

L'*key-id*élément de la clé ARNs des clés multirégionales commence par le `mrk-` préfixe. Voici un exemple d'ARN de clé pour une clé KMS multi-région.

```
arn:aws:kms:us-west-2:111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab
```

ID de clé

L'ID de clé identifie de manière unique une clé KMS au sein d'un compte et d'une région. Pour obtenir de l'aide sur la recherche de l'ID de clé d'une clé KMS, veuillez consulter the section called “Trouvez l'ID et l'ARN de la clé”.

Voici un exemple d'ID de clé pour une clé KMS de région unique.

```
1234abcd-12ab-34cd-56ef-1234567890ab
```

La clé IDs des clés multirégionales commence par le `mrk-` préfixe. Voici un exemple d'ID de clé pour une clé KMS multi-région.

```
mrk-1234abcd12ab34cd56ef1234567890ab
```

ARN d'alias

L'alias ARN est le nom Amazon Resource (ARN) d'un AWS KMS alias. Il s'agit d'un identifiant unique et complet pour l'alias et pour la clé KMS qu'il représente. Un ARN d'alias inclut Compte AWS la région et le nom de l'alias.

À tout moment, un ARN d'alias identifie une clé KMS particulière. Toutefois, comme vous pouvez modifier la clé KMS associée à l'alias, l'ARN d'alias peut identifier différentes clés KMS à des moments différents. Pour obtenir de l'aide sur la recherche de l'ARN d'alias d'une clé KMS, veuillez consulter Trouvez le nom d'alias et l'ARN de l'alias pour une clé KMS.

Le format d'un ARN d'alias est le suivant :

```
arn:<partition>:kms:<region>:<account-id>:alias/<alias-name>
```

Ce qui suit est l'ARN d'alias pour un `ExampleAlias` fictif.

```
arn:aws:kms:us-west-2:111122223333:alias/ExampleAlias
```

Nom d'alias

Le nom d'alias est une chaîne comportant jusqu'à 256 caractères. Il identifie de manière unique une clé KMS associée au sein d'un compte et d'une région. Dans l' AWS KMS API, les noms d'alias commencent toujours par `alias/`. Pour obtenir de l'aide sur la recherche du nom d'alias d'une clé KMS, veuillez consulter Trouvez le nom d'alias et l'ARN de l'alias pour une clé KMS.

Le format d'un nom d'alias est le suivant :

```
alias/<alias-name>
```

Par exemple :

```
alias/ExampleAlias
```

Le préfixe `aws/` d'un nom d'alias est réservé aux Clés gérées par AWS. Vous ne pouvez pas créer d'alias avec ce préfixe. Par exemple, le nom d'alias du Clé gérée par AWS pour Amazon Simple Storage Service (Amazon S3) est le suivant.

```
alias/aws/s3
```

Clés asymétriques AWS KMS

Une clé KMS asymétrique représente une paire de clés publiques et de clés privées mathématiquement liées entre elles. Vous pouvez donner la clé publique à n'importe qui, même s'il ne s'agit pas d'une personne de confiance, mais la clé privée doit rester secrète.

Dans une clé KMS asymétrique, la clé privée est créée AWS KMS et ne sort jamais AWS KMS non chiffrée. Pour utiliser la clé privée, vous devez appeler AWS KMS. Vous pouvez utiliser la clé publique qu'elle contient AWS KMS en appelant les opérations de l' AWS KMS API. Vous pouvez également télécharger la clé publique et l'utiliser en dehors de AWS KMS.

Si votre cas d'utilisation nécessite un chiffrement externe AWS par des utilisateurs qui ne peuvent pas appeler AWS KMS, les clés KMS asymétriques sont un bon choix. Toutefois, si vous créez une clé KMS pour chiffrer les données que vous stockez ou gérez dans un AWS service, utilisez une clé KMS de chiffrement symétrique. AWS les services intégrés AWS KMS utilisent uniquement des clés KMS de chiffrement symétriques pour chiffrer vos données. Ces services ne prennent pas en charge le chiffrement avec des clés KMS asymétriques.

AWS KMS prend en charge trois types de clés KMS asymétriques.

Clés RSA KMS

Une clé KMS avec une paire de clés RSA pour le chiffrement et le déchiffrement ou pour la signature et la vérification (mais pas les deux). AWS KMS prend en charge plusieurs longueurs de clé pour différentes exigences de sécurité.

Pour obtenir des informations techniques sur les algorithmes de chiffrement et de signature compatibles AWS KMS avec les clés RSA KMS, consultez les spécifications des clés RSA.

Clés KMS à courbe elliptique (ECC)

Une clé KMS avec une paire de clés à courbe elliptique pour la signature et la vérification ou pour obtenir des secrets partagés (mais pas les deux). AWS KMS prend en charge plusieurs courbes couramment utilisées.

Pour plus de détails techniques sur les algorithmes de signature compatibles AWS KMS avec les clés ECC KMS, voir Spécifications des clés à courbe elliptique.

Clés ML-DSA KMS

Une clé KMS avec une paire de clés ML-DSA pour la signature et la vérification. ML-DSA est une norme de cryptographie post-quantique développée par le National Institute of Standards and Technology (NIST) des États-Unis pour se protéger contre les menaces de sécurité posées par l'informatique quantique. ML-DSA est l'algorithme de signature numérique recommandé pour les entreprises qui passent des algorithmes de signature numérique RSA ou Elliptic Curve à la cryptographie sécurisée post-quantique.

AWS KMS prend en charge plusieurs longueurs de clé pour différentes exigences de sécurité. Pour obtenir des informations techniques sur les algorithmes de signature compatibles AWS KMS avec les clés KMS ML-DSA, reportez-vous à la section Spécification des clés ML-DSA.

SM2 Clés KMS (régions de Chine uniquement)

Une clé KMS avec une paire de SM2 clés pour le chiffrement et le déchiffrement, la signature et la vérification, ou pour la dérivation de secrets partagés (vous devez choisir un type d'utilisation de clé).

Pour obtenir des informations techniques sur les algorithmes de chiffrement et de signature compatibles AWS KMS avec les clés SM2 KMS (régions de Chine uniquement), consultez les spécifications des SM2 clés.

Pour obtenir de l'aide quant au choix de la configuration de votre clé asymétrique, veuillez consulter Choix du type de clé KMS à créer.

Régions

Les clés KMS asymétriques et les paires de clés de données asymétriques sont prises en charge dans tous Régions AWS les AWS KMS supports.

En savoir plus

- Pour créer des clés KMS asymétriques, veuillez consulter Créer une clé KMS asymétrique.
- Pour créer des clés KMS multi-région asymétriques, veuillez consulter Création de clés primaires multirégionales.
- Pour savoir comment signer des messages et vérifier des signatures à l'aide des clés KMS asymétriques, veuillez consulter Signature numérique avec la nouvelle fonction de clés asymétriques d' AWS KMS dans le blog de sécuritéAWS .
- Pour en savoir plus sur les considérations particulières relatives à la suppression de clés KMS asymétriques, consultezDeleting asymmetric KMS keys.
- Pour identifier et afficher les clés KMS asymétriques, consultezIdentifier les clés KMS asymétriques.

Clés HMAC entrées AWS KMS

Les clés KMS à code d'authentification des messages basé sur le hachage (HMAC) sont des clés symétriques que vous utilisez pour les générer et les vérifier. HMACs AWS KMS Les éléments de clé uniques associés à chaque clé KMS HMAC fournissent la clé secrète requise par les algorithmes HMAC. Vous pouvez utiliser une clé KMS HMAC avec les opérations `GenerateMac` et `VerifyMac` pour vérifier l'intégrité et l'authenticité des données dans AWS KMS.

Les algorithmes HMAC combinent une fonction de hachage cryptographique et une clé secrète partagée. Ils prennent un message et une clé secrète, comme les éléments de clé d'une clé KMS HMAC, et renvoient un code unique de taille fixe ou une balise. Si même un caractère du message change ou si la clé secrète n'est pas identique, la balise obtenue est entièrement différente. En exigeant une clé secrète, HMAC garantit également l'authenticité ; il est impossible de générer une étiquette HMAC identique sans la clé secrète. HMACs sont parfois appelées signatures symétriques, car elles fonctionnent comme des signatures numériques, mais utilisent une seule clé pour la signature et la vérification.

Les clés HMAC KMS et les algorithmes HMAC AWS KMS utilisés sont conformes aux normes industrielles définies dans la RFC 2104. L' AWS KMS GenerateMacopération génère des balises HMAC standard. Les clés HMAC KMS sont générées dans des modules de sécurité AWS KMS matériels certifiés dans le cadre du programme de validation des modules cryptographiques FIPS 140-3 (sauf dans les régions de Chine (Pékin) et de Chine (Ningxia)) et ne sont jamais non chiffrées. AWS KMS Pour utiliser une clé KMS HMAC, vous devez appeler AWS KMS.

Vous pouvez utiliser les clés KMS HMAC pour déterminer l'authenticité d'un message, comme un jeton Web JSON (JWT), des informations de carte de crédit tokenisée ou un mot de passe envoyé. Elles peuvent également être utilisées comme fonctions de dérivation de clés sécurisées (KDFs), en particulier dans les applications qui nécessitent des clés déterministes.

Les clés HMAC KMS offrent un avantage par rapport aux logiciels HMACs d'application, car le contenu clé est généré et utilisé entièrement dans le logiciel AWS KMS, sous réserve des contrôles d'accès que vous définissez sur la clé.

> **Tip**
>
> Les bonnes pratiques recommandent de limiter la durée pendant laquelle tout mécanisme de signature, y compris un HMAC, est effectif. Cela dissuade une attaque où l'acteur utilise un message signé pour établir la validité à plusieurs reprises ou longtemps après le remplacement du message. Les balises HMAC n'incluent pas d'horodatage, mais vous pouvez inclure un horodatage dans le jeton ou le message pour vous aider à détecter le moment où il convient d'actualiser le HMAC.

Opérations cryptographiques prises en charge

Les clés HMAC KMS prennent uniquement en charge les opérations de chiffrement `GenerateMac` et `VerifyMac`. Vous ne pouvez pas utiliser de clés KMS HMAC pour chiffrer des données ou signer des messages, ni pour utiliser tout autre type de clé KMS dans les opérations HMAC. Lorsque vous utilisez l'opération `GenerateMac`, vous fournissez un message pouvant atteindre 4 096 octets, une clé KMS HMAC et l'algorithme MAC compatible avec la spécification de clé HMAC, tandis que `GenerateMac` calcule la balise HMAC. Pour vérifier une balise HMAC, vous devez fournir la balise HMAC, ainsi que le même message, la clé KMS HMAC et l'algorithme MAC que `GenerateMac` a utilisé pour calculer la balise HMAC d'origine. L'opération `VerifyMac` calcule la balise HMAC et vérifie qu'elle est identique à la balise HMAC fournie. Si les balises HMAC en entrée et calculées ne sont pas identiques, la vérification échoue.

Les clés KMS HMAC ne prennent pas en charge la rotation automatique des clés et vous ne pouvez pas créer de clé KMS HMAC dans un magasin de clés personnalisé.

Si vous créez une clé KMS pour chiffrer les données d'un AWS service, utilisez une clé de chiffrement symétrique. Vous ne pouvez pas utiliser de clé KMS HMAC.

Régions

Les clés HMAC KMS sont prises en charge dans tous Régions AWS les AWS KMS supports.

En savoir plus

- Pour créer des clés HMAC KMS, voirCréer une clé KMS HMAC.
- Pour créer des clés HMAC KMS multirégionales, consultez. Clés multirégionales dans AWS KMS
- Pour examiner la différence entre la politique de clé par défaut définie par la AWS KMS console pour les clés HMAC KMS, consultezthe section called “Permet aux utilisateurs de clés d'utiliser une clé KMS pour les opérations de chiffrement”.
- Pour identifier et afficher les clés HMAC KMS, consultezIdentifier les clés HMAC KMS.
- Pour en savoir plus sur l'utilisation HMACs de jetons Web JSON pour créer des jetons Web JSON, consultez la section Comment protéger HMACs l'intérieur AWS KMS dans le blog sur la AWS sécurité.
- Écoutez un podcast : Présentation HMACs de AWS Key Management Service on The Official AWS Podcast.

Clés ML-DSA entrées AWS KMS

AWS Key Management Service (AWS KMS) prend en charge l'algorithme de signature numérique Module-Lattice (ML-DSA) pour les signatures cryptographiques post-quantiques. Cette mise en œuvre est conforme à la norme Federal Information Processing Standards (FIPS) 204 afin de contribuer à la protection contre les futures menaces informatiques quantiques. AWS KMS crée et protège toutes les clés ML-DSA et les opérations de signature dans les modules de sécurité matériels validés FIPS 140-3 Security Level 3. Pour trouver un équilibre entre sécurité et performances, ML-DSA AWS KMS propose trois niveaux de sécurité distincts grâce à différentes spécifications clés, ML_DSA_44, ML_DSA_65 et ML_DSA_87.

AWS KMS prend en charge les signatures de clé asymétriques pour les messages d'une taille maximale de 4 Ko en utilisant le type de `RAW` message. Pour les messages plus volumineux, vous

devez calculer en externe la représentation du message de 64 octets μ utilisée pour la signature ML-DSA, comme défini dans la section 6.2 de la norme NIST FIPS 204. Utilisez le type de `EXTERNAL_MU` message dans l'opération de AWS KMS signature pour spécifier ce message prétraité de 64 octets. Les signatures produites par le μ calculé en externe sont les mêmes que `RAW` celles produites lors de l'utilisation du même message et de la même clé privée. Notez que cette signature est différente du « pré-hachage » ML-DSA ou du HashML-DSA de la section 5.4 du NIST FIPS 204.

Pour plus d'informations sur l'utilisation de ML-DSA et du type de message EXTERNAL_MU, consultez. Caractéristiques clés de la ML-DSA

Pour un exemple d'utilisation de ML-DSA et du type de message EXTERNAL_MU, consultez. Vérification hors ligne avec des paires de clés ML-DSA

Clés multirégionales dans AWS KMS

AWS KMS prend en charge les clés multirégionales, qui sont AWS KMS keys différentes Régions AWS et peuvent être utilisées de manière interchangeable, comme si vous aviez la même clé dans plusieurs régions. Chaque ensemble de clés multirégionales associées possède le même contenu clé et le même identifiant de clé. Vous pouvez donc chiffrer les données dans une clé Région AWS et les déchiffrer dans une autre Région AWS sans les chiffrer à nouveau ou sans passer d'appel interrégional à. AWS KMS

Comme toutes les clés KMS, les clés multirégionales ne sont jamais AWS KMS déchiffrées. Vous pouvez créer des clés multi-région symétriques ou asymétriques pour le chiffrement ou la signature, créer des clés multi-régions HMAC pour la génération et la vérification de balises HMAC, mais aussi créer des clés multi-région avec des éléments de clé importés ou des éléments de clé générés par AWS KMS . Vous devez gérer chaque clé multi-région indépendamment, notamment en créant des alias et des balises, en définissant les politiques et les octrois de clé, en les activant et en les désactivant de manière sélective. Vous pouvez utiliser des clés multi-région dans le cadre de toutes les opérations de chiffrement que vous pouvez effectuer avec des clés à région unique.

Les clés multi-région sont une solution flexible et puissante pour de nombreux scénarios courants liés à la sécurité des données.

Reprise après sinistre

> Dans une architecture de sauvegarde et de restauration, les clés multirégionales vous permettent de traiter les données chiffrées sans interruption, même en cas de Région AWS panne. Les données conservées dans les régions de sauvegarde peuvent être déchiffrées dans la région de

sauvegarde, et les données nouvellement chiffrées dans la région de sauvegarde peuvent être déchiffrées dans la région principale lorsque cette région est restaurée.

Gestion globale des données

Les entreprises qui opèrent à l'échelle mondiale ont besoin de données distribuées dans le monde entier et qui soient disponibles entre les Régions AWS. Vous pouvez créer des clés multi-région dans toutes les régions où résident vos données, puis utiliser les clés comme s'il s'agissait d'une clé à région unique sans la latence d'un appel inter-régions ou le coût du re-chiffrement des données sous une clé différente dans chaque région.

Applications de signature distribuées

Les applications qui nécessitent des fonctionnalités de signature inter-régions peuvent utiliser des clés de signature asymétriques multi-région pour générer des signatures numériques identiques de manière cohérente et répétée dans différentes Régions AWS.

Si vous utilisez le chaînage de certificats avec un magasin de confiance global unique (pour une autorité de certification (CA) racine unique, et un intermédiaire régional CAs signé par l'autorité de certification racine, vous n'avez pas besoin de clés multirégionales. Toutefois, si votre système ne prend pas en charge les protocoles intermédiaires CAs, tels que la signature d'applications, vous pouvez utiliser des clés multirégionales pour garantir la cohérence des certifications régionales.

Applications active/active couvrant plusieurs régions

Certaines charges de travail et applications peuvent couvrir plusieurs régions dans des architectures active/active. Pour ces applications, les clés multi-région peuvent réduire la complexité en fournissant les mêmes éléments de clé pour les opérations simultanées de chiffrement et de déchiffrement sur les données susceptibles de se déplacer au-delà des limites de la région.

Vous pouvez utiliser des clés multirégionales avec des bibliothèques de chiffrement côté client, telles que le AWS Encryption SDKSDK de chiffrement de AWS base de données et le chiffrement côté client Amazon S3.

AWS les services intégrés au chiffrement au repos ou aux AWS KMS signatures numériques traitent actuellement les clés multirégionales comme s'il s'agissait de clés à région unique. Ils peuvent ré-encapsulper ou re-chiffrer les données déplacées entre les régions. Par exemple, la réplication inter-régions Amazon S3 déchiffre et rechiffre les données sous une clé KMS dans la région de destination, même lors de la réplication d'objets protégés par une clé multi-région.

Les clés multi-région ne sont pas globales. Vous créez une clé principale multi-région, puis vous la répliquez dans les régions que vous sélectionnez dans une partition AWS. Vous gérez ensuite la clé multi-région dans chaque région de manière indépendante. Ni AWS ne AWS KMS crée ni ne réplique automatiquement les clés multirégionales dans aucune région en votre nom. Clés gérées par AWS, les clés KMS que les AWS services créent pour vous dans votre compte sont toujours des clés à région unique.

Dans les régions de Chine, vous pouvez utiliser la fonctionnalité de clé multirégionale pour répliquer les clés KMS dans la partition des régions de Chine (`aws-cn`). Par exemple, vous pouvez répliquer une clé de la région Chine (Pékin) vers la région Chine (Ningxia), ou inversement. En répliquant une clé d'une région de Chine à une autre, vous acceptez d'utiliser celle AWS Key Management Service de la région de destination et de respecter toutes les conditions d'accord applicables à la région de destination. Vous ne pouvez pas répliquer une clé des régions de Pékin et du Ningxia dans une AWS région située en dehors de la partition des régions de Chine. De même, vous ne pouvez pas répliquer une clé d'une région située en dehors de la partition des régions de Chine vers les régions de Pékin et de Ningxia.

Vous ne pouvez pas transformer une clé à région unique en clé multi-région. Cette conception garantit que toutes les données protégées avec les clés à région unique existantes conservent les mêmes propriétés de résidence et de souveraineté des données.

Pour la plupart des besoins de sécurité des données, l'isolation régionale et la tolérance aux pannes des ressources régionales font des clés AWS KMS unirégionales standard la solution la mieux adaptée. Toutefois, lorsque vous avez besoin de chiffrer ou de signer des données dans des applications côté client entre plusieurs régions, les clés multi-région peuvent être la solution.

Régions

Les clés multirégionales sont prises en charge dans tous Régions AWS les AWS KMS supports.

Tarification et quotas

Chaque clé d'un ensemble de clés multi-région associées compte comme une clé KMS pour la tarification et les quotas. Les quotas AWS KMS sont calculés séparément pour chaque région d'un compte. L'utilisation et la gestion des clés multi-région dans chaque région sont prises en compte dans les quotas pour cette région.

Types de clés KMS non pris en charge

Vous pouvez créer les types suivants de clés KMS multirégions :

- Clés KMS de chiffrement symétrique
- Clés KMS asymétriques
- Clés KMS HMAC
- Clés KMS avec des éléments de clé importés

Vous ne pouvez pas créer de clés multi-région dans un magasin de clés personnalisé.

En savoir plus

- Pour savoir comment contrôler l'accès aux clés KMS multirégionales, consultezContrôler l'accès aux clés multirégionales.
- Pour créer des clés KMS primaires multirégionales de n'importe quel type, consultezCréation de clés primaires multirégionales.
- Pour créer des répliques de clés KMS multirégionales, consultezCréation de répliques de clés multirégionales.
- Pour mettre à jour la région principale, voirModifier la clé primaire dans un ensemble de clés multirégionales.
- Pour identifier et afficher les clés KMS multirégionales, consultezIdentifier les clés HMAC KMS.
- Pour en savoir plus sur les considérations particulières relatives à la suppression de clés KMS multirégionales, consultezDeleting multi-Region keys.

Terminologie et concepts

Les termes et concepts suivants sont utilisés avec des clés multi-région.

Clé multi-région

Une clé multi-région fait partie d'un ensemble de clés KMS avec le même ID de clé et les mêmes éléments de clé (et d'autres propriétés partagées) dans différentes Régions AWS. Chaque clé multi-région est une clé KMS pleinement fonctionnelle qui peut être utilisée indépendamment des clés multi-région associées. Comme toutes les clés multirégionales associées ont le même identifiant de clé et le même contenu clé, elles sont interopérables, c'est-à-dire que toute clé multirégionale associée Région AWS peut déchiffrer le texte chiffré par n'importe quelle autre clé multirégionale associée.

Vous définissez la propriété multi-région d'une clé KMS lors de sa création. Vous ne pouvez pas modifier propriété multi-région sur une clé existante. Vous ne pouvez pas convertir une clé à région unique en clé multi-région ou convertir une clé multi-région en clé à région unique. Pour déplacer des charges de travail existantes dans des scénarios multi-région, vous devez chiffrer à nouveau vos données ou créer de nouvelles signatures avec de nouvelles clés multi-région.

Une clé multirégionale peut être symétrique ou asymétrique et elle peut utiliser du matériel clé ou du matériel AWS KMS clé importé. Vous ne pouvez pas créer de clés multi-région dans un magasin de clés personnalisé.

Dans un ensemble de clés multi-région associées, il y a exactement une clé principale à tout moment. Vous pouvez créer des clés de réplica de cette clé principale dans d'autres Régions AWS. Vous pouvez également mettre à jour la région principale, qui transforme la clé principale en clé de réplica et transforme une clé de réplica spécifiée en clé principale. Toutefois, vous ne pouvez conserver qu'une seule clé primaire ou une seule clé de réplique dans chacune d'elles Région AWS. Toutes les régions doivent être dans la même partition AWS.

Vous pouvez avoir plusieurs ensembles de clés multi-région associées dans la même ou dans plusieurs Régions AWS. Bien que les clés multi-région associées soient interopérables, les clés multi-région non associées ne sont pas interopérables.

Clé primaire

Une clé primaire multirégionale est une clé KMS qui peut être répliquée Régions AWS dans d'autres clés de la même partition. Chaque ensemble de clés multi-région ne possède qu'une seule clé principale.

Une clé principale diffère d'une clé de réplica comme suit :

- Seule une clé principale peut être répliquée.
- La clé principale est la source de propriétés partagées de ses clés de réplica, y compris les éléments de clé et l'ID de clé.
- Vous pouvez activer et désactiver la rotation automatique des clés seulement sur une clé principale.
- Vous pouvez planifier la suppression d'une clé principale à tout moment. Mais il ne AWS KMS supprimera pas une clé primaire tant que toutes ses clés répliques ne seront pas supprimées.

Cependant, les clés principales et les clés de réplica ne diffèrent pas au niveau des propriétés cryptographiques. Vous pouvez utiliser une clé primaire et ses clés de réplica de manière interchangeable.

Vous n'êtes pas tenu de répliquer une clé principale. Vous pouvez l'utiliser comme n'importe quelle clé KMS et la répliquer si elle est utile. Toutefois, étant donné que les clés multi-région ont des propriétés de sécurité différentes de celles des clés à région unique, nous vous recommandons de créer une clé multi-région uniquement lorsque vous envisagez de la répliquer.

Clé de réplica

Une clé de réplication multirégionale est une clé KMS qui possède le même identifiant et le même matériau de clé que sa clé primaire et les clés de réplique associées, mais qui existe dans une autre Région AWS.

Une clé de réplica est une clé KMS pleinement fonctionnelle avec ses propres politiques de clé, octrois, alias, balises et autres propriétés. Il ne s'agit pas d'une copie ou d'un pointeur vers la clé principale ou toute autre clé. Vous pouvez utiliser une clé de réplica même si sa clé principale et toutes les clés de réplica associées sont désactivées. Vous pouvez également transformer une clé de réplica en clé principale et une clé principale en clé de réplica. Une fois créée, une clé de réplica s'appuie sur sa clé principale uniquement pour la rotation des clés et la mise à jour de la région principale.

Les clés principales et les clés de réplica ne diffèrent pas au niveau des propriétés cryptographiques. Vous pouvez utiliser une clé primaire et ses clés de réplica de manière interchangeable. Les données chiffrées par une clé principale ou de réplica peuvent être déchiffrées par la même clé, ou par toute clé principale ou de réplica associée.

Répliquer

Vous pouvez répliquer une clé primaire multirégionale dans une autre Région AWS clé de la même partition. Lorsque vous le faites, AWS KMS crée une clé de réplique multirégionale dans la région spécifiée avec le même ID de clé et d'autres propriétés partagées que sa clé primaire. Ensuite, il transporte en toute sécurité les éléments de clé au-delà des limites de la région et les associe à la nouvelle clé de réplica, le tout dans AWS KMS.

Propriétés partagées

Les propriétés partagées sont les propriétés d'une clé primaire multirégionale qui sont partagées avec ses clés de réplique. AWS KMS crée les clés de réplique avec les mêmes valeurs de propriétés partagées que celles de la clé primaire. Ensuite, il synchronise périodiquement les valeurs de

propriété partagées de la clé principale avec ses clés de réplica. Vous ne pouvez pas définir ces propriétés sur une clé de réplica.

Voici les propriétés partagées des clés multi-région.

- ID de clé — (L'élément `Region` de l'ARN de clé diffère.)
- Éléments de clé
- Origine des éléments de clé
- Spécifications des clés et algorithmes de chiffrement
- Utilisation de la clé
- Rotation automatique des clés — Vous pouvez activer et désactiver la rotation automatique des clés uniquement sur la clé principale. De nouvelles clés de réplica sont créées avec toutes les versions des éléments de clé partagés. Pour plus de détails, veuillez consulter Rotating multi-Region keys.
- Rotation à la demande : vous pouvez effectuer une rotation à la demande uniquement sur la clé primaire. De nouvelles clés de réplica sont créées avec toutes les versions des éléments de clé partagés. Pour plus de détails, veuillez consulter Rotating multi-Region keys.

Vous pouvez également considérer les désignations principales et de réplica des clés multi-région associées comme des propriétés partagées. Lorsque vous créez de nouvelles clés répliquées ou que vous mettez à jour la clé primaire, la modification est AWS KMS synchronisée avec toutes les clés multirégionales associées. Lorsque ces modifications sont terminées, toutes les clés multi-région associées répertorient avec précision leur clé principale et leurs clés de réplica.

Toutes les autres propriétés des clés multi-région sont des propriétés indépendantes, y compris la description, la politique de clé, les octrois, les états de clé activé et désactivé, les alias et les balises. Vous pouvez définir les mêmes valeurs pour ces propriétés sur toutes les clés multi-région associées, mais si vous modifiez la valeur d'une propriété indépendante, AWS KMS ne la synchronise pas.

Vous pouvez suivre la synchronisation des propriétés partagées de vos clés multi-région. Dans votre AWS CloudTrail journal, recherchez l'SynchronizeMultiRegionKeyévénement.

Considérations sur la sécurité pour les clés multi-région

Utilisez une clé AWS KMS multirégionale uniquement lorsque vous en avez besoin. Les clés multi-région fournissent une solution flexible et évolutive pour les applications qui déplacent des données chiffrées entre Régions AWS ou ont besoin d'un accès inter-régions. Envisagez une clé multi-région

si vous devez partager, déplacer ou sauvegarder des données protégées entre les régions ou si vous avez besoin de créer des signatures numériques identiques d'applications fonctionnant dans différentes régions.

Toutefois, le processus de création d'une clé multi-région déplace vos éléments de clé au-delà des frontières des Région AWS au sein de AWS KMS. Le texte chiffré généré par une clé multi-région peut potentiellement être déchiffré par plusieurs clés associées dans plusieurs emplacements géographiques. Il y a également des avantages importants pour les services et les ressources isolés dans la région. Chaque Région AWS est indépendante et isolée des autres régions. Les régions fournissent une tolérance aux pannes, une stabilité et une résilience, et peuvent également réduire la latence. Elles vous permettent de créer des ressources redondantes qui restent disponibles et qui ne sont pas affectées par les pannes dans les autres régions. En AWS KMS, ils garantissent également que chaque texte chiffré peut être déchiffré par une seule clé.

Les clés multi-région soulèvent également de nouvelles considérations de sécurité :

- Le contrôle de l'accès et l'application de la politique de sécurité des données sont plus complexes avec les clés multi-région. Vous devez vous assurer que la politique est auditée de manière cohérente sur la clé dans plusieurs régions isolées. Vous devez également utiliser la politique pour appliquer les frontières, au lieu de vous appuyer sur des clés séparées.

  Par exemple, vous devez définir des conditions de politique sur les données pour empêcher les équipes de paie d'une région de lire les données de paie pour une autre région. En outre, vous devez utiliser le contrôle d'accès pour empêcher un scénario dans lequel une clé multi-région dans une région protège les données d'un locataire et une clé multi-région associée dans une autre région protège les données d'un autre locataire.
- L'audit des clés entre les régions est également plus complexe. Avec les clés multi-région, vous devez examiner et réconcilier les activités d'audit entre plusieurs régions afin d'obtenir une compréhension complète des activités de clé liées aux données protégées.
- La conformité aux mandats de résidence des données peut être plus complexe. Avec les régions isolées, vous pouvez garantir la résidence des données et la conformité à la souveraineté des données. Les clés KMS d'une région donnée peuvent déchiffrer des données sensibles uniquement dans cette région. Les données chiffrées dans une région peuvent rester complètement protégées et inaccessibles dans toute autre région.

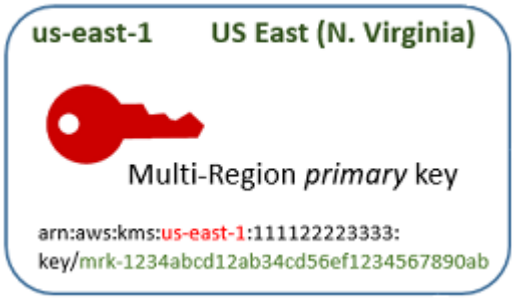
  Pour vérifier la résidence et la souveraineté des données à l'aide de clés multirégionales, vous devez mettre en œuvre des politiques d'accès et compiler AWS CloudTrail des événements dans plusieurs régions.

Pour faciliter la gestion du contrôle d'accès sur les clés multirégionales, l'autorisation de répliquer une clé multirégionale (kms : ReplicateKey) est distincte de l'autorisation standard de création de clés (kms :). CreateKey AWS KMS Prend également en charge plusieurs conditions de politique pour les clés multirégionaleskms:MultiRegion, notamment celles qui autorisent ou refusent l'autorisation de créer, d'utiliser ou de gérer des clés multirégionales et kms:ReplicaRegion qui restreint les régions dans lesquelles une clé multirégionale peut être répliquée. Pour plus de détails, veuillez consulter Contrôler l'accès aux clés multirégionales.

Fonctionnement des clés multi-région

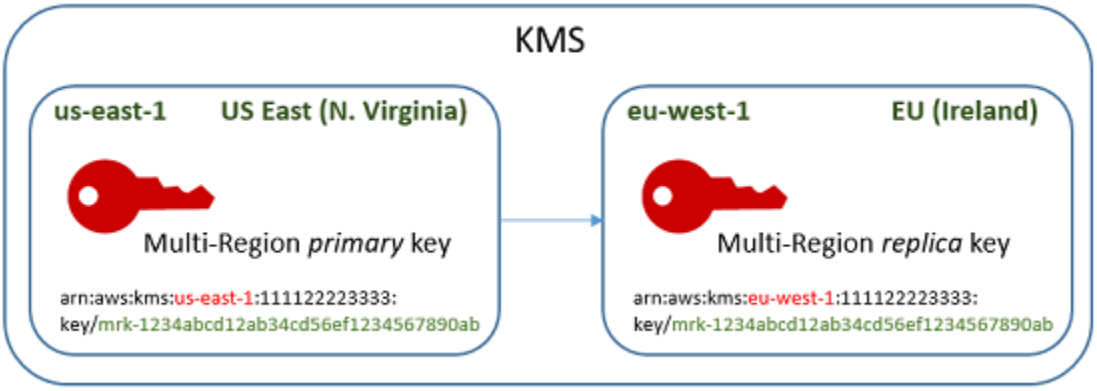
Vous commencez par créer une clé primaire multirégionale symétrique ou asymétrique dans une clé AWS KMS compatible, telle Région AWS que USA East (Virginie du Nord). Vous décidez si une clé est à région unique ou multi-région uniquement lorsque vous la créez ; vous ne pouvez pas modifier cette propriété ultérieurement. Comme pour toute clé KMS, vous définissez une politique de clé pour la clé multi-région, et vous pouvez créer des octrois et ajouter des alias et des balises pour la catégorisation et l'autorisation. (Ce sont des propriétés indépendantes qui ne sont pas partagées ou synchronisées avec d'autres clés.) Vous pouvez utiliser votre clé principale multi-région dans les opérations de chiffrement pour le chiffrement ou la signature.

Vous pouvez créer une clé primaire multirégionale dans la AWS KMS console ou en utilisant l'CreateKeyAPI avec le MultiRegion paramètre défini sur. true Notez que les clés multi-région ont un ID de clé distinctif qui commence par mrk-. Vous pouvez utiliser le mrk- préfixe pour vous identifier MRKs par programmation.



Si vous le souhaitez, vous pouvez répliquer la clé primaire multirégionale dans une ou plusieurs Régions AWS autres clés de la même AWS partition, par exemple en Europe (Irlande). Lorsque vous le faites, AWS KMS crée une réplique de clé dans la région spécifiée avec le même ID de clé et d'autres propriétés partagées que la clé primaire. Ensuite, il transporte en toute sécurité les éléments de clé au-delà des limites de la région et les associe à la nouvelle clé KMS dans la région de destination, le tout dans AWS KMS. Le résultat donne deux clés multi-région associées : une clé principale et une clé de réplica, pouvant être utilisées de manière interchangeable.

Vous pouvez créer une réplique de clé multirégionale dans la AWS KMS console ou à l'aide de l'ReplicateKeyAPI.



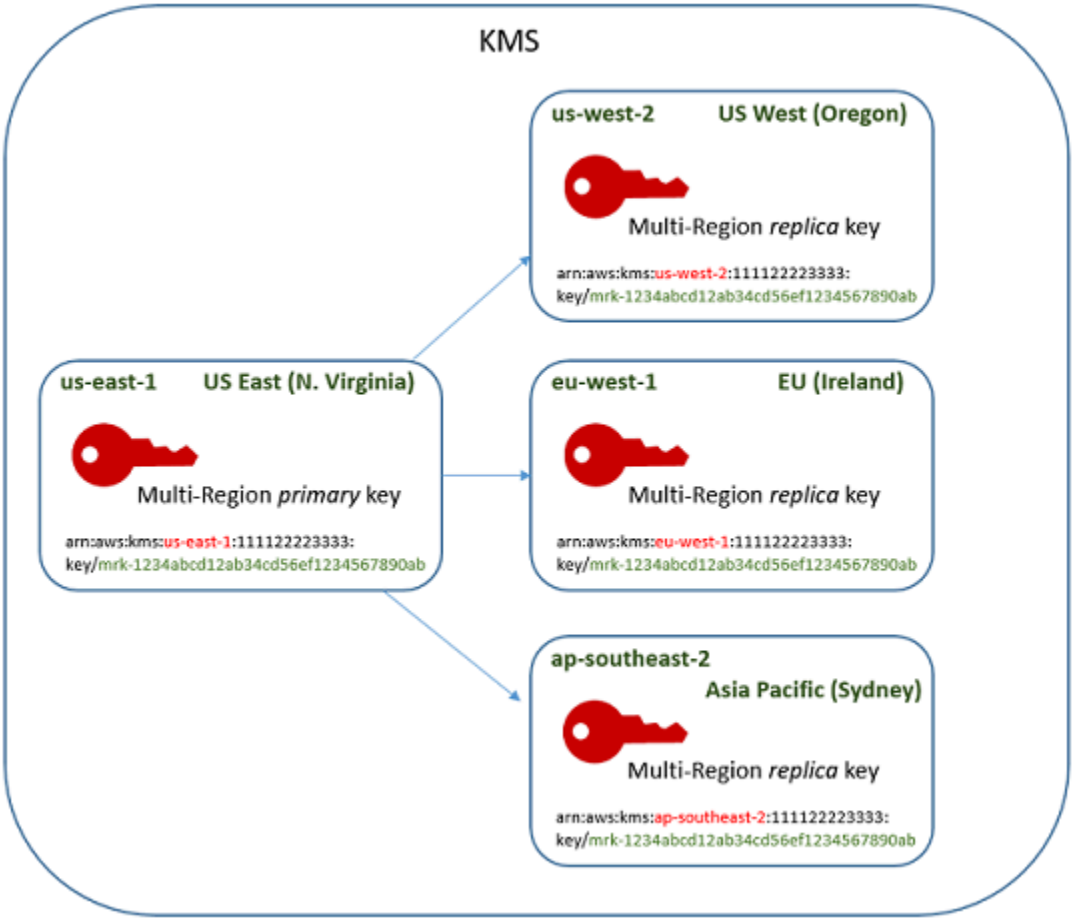
La clé de réplica multi-région qui en résulte est une clé KMS pleinement fonctionnelle, avec les mêmes propriétés partagées que clé principale. À tous autres égards, il s'agit d'une clé KMS indépendante avec ses propres description, politique de clé, octrois, alias et balises. L'activation ou la désactivation d'une clé multi-région n'a aucun effet sur les clés multi-région associées. Vous pouvez utiliser les clés principales et de réplica indépendamment dans les opérations cryptographiques ou coordonner leur utilisation. Par exemple, vous pouvez chiffrer des données avec la clé principale dans la région USA Est (Virginie du Nord), déplacer les données vers la région UE (Irlande) et utiliser la clé de réplica pour déchiffrer les données.

Les clés multi-région associées ont le même ID de clé. Leur clé ARNs (Amazon Resource Names) ne diffère que dans le champ Région. Par exemple, la clé primaire multirégionale et les clés de réplique peuvent avoir l'exemple de clé ARNs suivant. L'ID de clé (le dernier élément de l'ARN de clé) est identique. Les deux clés ont l'ID de clé distinctif des clés multi-régions, qui commence par mrk-.

```
Primary key: arn:aws:kms:us-
east-1:111122223333:key/mrk-1234abcd12ab34cd56ef12345678990ab
Replica key: arn:aws:kms:eu-
west-1:111122223333:key/mrk-1234abcd12ab34cd56ef12345678990ab
```

Le même ID de clé est requis pour l'interopérabilité. Lors du chiffrement, AWS KMS lie l'ID de clé KMS au texte chiffré afin que le texte chiffré ne puisse être déchiffré qu'avec cette clé KMS ou une clé KMS avec le même identifiant de clé. Cette fonction facilite également la reconnaissance des clés multi-région associées ainsi que leur utilisation interchangeable. Par exemple, lorsque vous les utilisez dans une application, vous pouvez faire référence aux clés multi-région associées par leur ID de clé partagé. Ensuite, si nécessaire, spécifiez la région ou l'ARN pour les distinguer.

À mesure que vos besoins en matière de données évoluent, vous pouvez répliquer la clé primaire vers d'autres Régions AWS utilisateurs de la même partition, par exemple dans l'ouest des États-Unis (Oregon) et dans la région Asie-Pacifique (Sydney). Le résultat est quatre clés multirégionales associées avec le même matériau clé et la même clé IDs, comme indiqué dans le schéma suivant. Vous gérez les clés indépendamment. Vous pouvez les utiliser indépendamment ou de manière coordonnée. Par exemple, vous pouvez chiffrer des données avec la clé de réplica dans la région Asie-Pacifique (Sydney), déplacer les données vers la région USA Ouest (Oregon) et les déchiffrer avec la clé de réplica dans la région USA Ouest (Oregon).



Voici d'autres considérations pour les clés multi-région.

Synchronisation des propriétés partagées : si une propriété partagée des clés multirégionales change, la modification est AWS KMS automatiquement synchronisée entre la clé primaire et toutes ses clés répliquées. Vous ne pouvez pas demander ou forcer la synchronisation des propriétés partagées. AWS KMS détecte et synchronise toutes les modifications pour vous. Vous pouvez toutefois auditer la synchronisation en utilisant l'SynchronizeMultiRegionKeyévénement dans CloudTrail les journaux.

Par exemple, si vous activez la rotation automatique des clés sur une clé primaire multirégionale symétrique, AWS KMS copie ce paramètre sur toutes ses clés répliques. Lorsque les éléments de clé sont soumis à une rotation, la rotation est synchronisée entre toutes les clés multi-région associées, de sorte qu'elles continuent d'avoir les mêmes éléments de clé actuels et d'accéder à toutes les versions plus anciennes des éléments de clé. Si vous créez une nouvelle clé de réplica, elle dispose des mêmes éléments de clé actuels que toutes les clés multi-région associées et de l'accès à toutes les versions précédentes des éléments de clé. Pour plus de détails, veuillez consulter Rotating multi-Region keys.

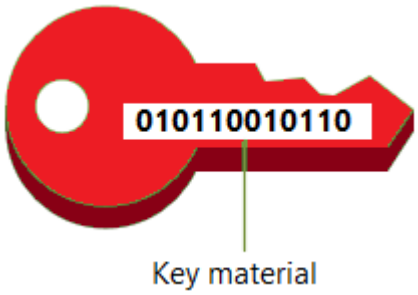
Modification de la clé principale — Chaque ensemble de clés multi-région doit avoir exactement une clé principale. La clé principale est la seule clé qui peut être répliquée. C'est également la source des propriétés partagées de ses clés de réplica. Toutefois, vous pouvez transformerr la clé principale en un réplica et transformer l'une des clés de réplica en clé principale. Vous pouvez procéder ainsi afin de supprimer une clé principale multi-région d'une région particulière ou de placer la clé principale dans une région plus proche des administrateurs de projet. Pour plus de détails, veuillez consulter Modifier la clé primaire dans un ensemble de clés multirégionales.

Suppression des clés multirégionales — Comme toutes les clés KMS, vous devez planifier la suppression des clés multirégionales avant de les AWS KMS supprimer. Lorsque la clé est en attente de suppression, vous ne pouvez pas l'utiliser dans une opération de chiffrement. Toutefois, une clé primaire multirégionale ne AWS KMS sera pas supprimée tant que toutes ses clés répliquées ne seront pas supprimées. Pour en savoir plus, consultez Deleting multi-Region keys.

Importation de matériel clé pour les AWS KMS clés

Vous pouvez créer une AWS KMS keys (clé KMS) avec le matériel clé que vous fournissez.

Une clé KMS est une représentation logique d'une clé de données. Les métadonnées d'une clé KMS incluent l'identifiant du matériel clé utilisé pour effectuer des opérations cryptographiques. Lorsque vous créez une clé KMS, par défaut, le matériel clé pour cette clé KMS est AWS KMS généré. Mais vous pouvez créer une clé KMS sans éléments de clé, puis importer vos propres éléments de clé dans cette clé KMS, une fonction souvent appelée « Bring Your Own Key » (BYOK).



> **Note**
>
> AWS KMS ne prend pas en charge le déchiffrement d'un AWS KMS texte chiffré par une clé KMS de chiffrement symétrique en dehors de AWS KMS, même si le texte chiffré a été chiffré sous une clé KMS avec du matériel clé importé. AWS KMS ne publie pas le format de texte chiffré requis par cette tâche, et le format peut changer sans préavis.

Lorsque vous utilisez du matériel clé importé, vous restez responsable du matériel clé tout en autorisant l'utilisation AWS KMS d'une copie de celui-ci. Vous pouvez choisir de le faire pour une ou plusieurs des raisons suivantes :

- Pour prouver que l'élément de clé a été généré en utilisant une source d'entropie correspondant à vos besoins.
- Pour utiliser le matériel clé de votre propre infrastructure avec AWS des services, et AWS KMS pour gérer le cycle de vie du matériel clé qu'il contient AWS.
- Pour utiliser des clés existantes et bien établies AWS KMS, telles que les clés pour la signature de code, la signature de certificats PKI et les applications associées à des certificats
- Pour définir une date d'expiration pour le contenu clé AWS et le supprimer manuellement, mais aussi pour le rendre à nouveau disponible à l'avenir. En revanche, une planification de suppression de clé nécessite une période d'attente de 7 à 30 jours, après laquelle vous ne pouvez pas récupérer la clé KMS supprimée.
- Posséder la copie originale du matériel clé et la conserver à l'extérieur AWS pour une durabilité accrue et une reprise après sinistre pendant tout le cycle de vie du matériau clé.
- Pour les clés asymétriques et les clés HMAC, l'importation crée des clés compatibles et interopérables qui fonctionnent à l'intérieur et à l'extérieur de. AWS

Types de clés KMS non pris en charge

AWS KMS prend en charge le matériel clé importé pour les types de clés KMS suivants. Vous ne pouvez pas importer des éléments de clé dans des clés KMS d'un magasin de clés personnalisé.

- Clés KMS de chiffrement symétrique
- Clés KMS asymétriques (sauf les clés ML-DSA)
- Clés KMS HMAC
- Clés multi-région de tous les types pris en charge.

Régions

Le matériel clé importé est pris en charge dans tous Régions AWS les AWS KMS supports.

Dans les régions de Chine, les principales exigences matérielles pour les clés KMS de chiffrement symétriques diffèrent de celles des autres régions. Pour en savoir plus, consultez Étape 3 : Chiffrement des éléments de clé.

En savoir plus

- Pour créer des clés KMS avec du matériel clé importé, voirCréation d'une clé KMS avec du matériel clé importé.
- Pour créer une alarme qui vous avertit lorsque le contenu clé importé d'une clé KMS approche de sa date d'expiration, voirCréer une CloudWatch alarme en cas d'expiration du matériel clé importé.
- Pour réimporter du contenu clé dans une clé KMS, consultezRéimportez le matériel clé.
- Pour importer de nouveaux éléments clés dans une clé KMS en vue d'une rotation à la demande, reportez-vous Importer de nouveaux documents clés aux sections etEffectuez une rotation des touches à la demande.
- Pour identifier et afficher les clés KMS contenant du matériel clé importé, voirIdentifiez les clés KMS avec du matériel clé importé.
- Pour en savoir plus sur les considérations particulières relatives à la suppression de clés KMS contenant du matériel clé importé, voirDeleting KMS keys with imported key material.

Considérations spéciales relatives au matériel clé importé

Avant de décider d'importer du matériel clé dans AWS KMS, vous devez comprendre les caractéristiques suivantes du matériel clé importé.

Vous générez les éléments de clé.

Vous êtes responsable de la génération des éléments de clé à l'aide d'une source aléatoire qui répond à vos exigences de sécurité.

Vous êtes responsable de la disponibilité et de la durabilité.

AWS KMS est conçu pour maintenir la haute disponibilité du matériel clé importé. Mais AWS KMS ne maintient pas la durabilité du matériau clé importé au même niveau que le matériau clé qui en AWS KMS génère. Pour en savoir plus, consultez Suppression des éléments de clé importés.

Vous pouvez supprimer les éléments de clé.

Vous pouvez supprimer les éléments de clé importés d'une clé KMS, ce qui la rend immédiatement inutilisable. De plus, lorsque vous importez des éléments de clé dans une clé KMS, vous pouvez définir sa date d'expiration si vous souhaitez qu'elle en ait une. Lorsque le délai d'expiration arrive, le AWS KMS matériel clé est supprimé. Sans éléments de clé, la clé KMS ne peut pas être utilisée dans une opération de chiffrement. Pour restaurer la clé, vous devez y réimporter les mêmes éléments de clé.

Vous ne pouvez pas modifier le matériau des clés pour les clés asymétriques, HMAC et multirégionales

Lorsque vous importez des éléments de clé dans une clé KMS, celle-ci est définitivement associée à ces éléments de clé. Vous pouvez réimporter les mêmes éléments de clé, mais vous ne pouvez pas en importer d'autres dans cette clé KMS. De plus, vous ne pouvez pas activer la rotation automatique des clés pour une clé KMS dont les éléments de clé sont importés. Toutefois, vous pouvez soumettre à une rotation manuelle une clé KMS avec les éléments de clé importés.

Vous pouvez effectuer une rotation à la demande sur des clés de chiffrement symétriques à région unique

Les clés de chiffrement symétriques à région unique avec du matériel clé importé prennent en charge la rotation à la demande. Vous pouvez importer plusieurs éléments clés dans ces clés et utiliser la rotation à la demande pour mettre à jour le contenu clé actuel. Le matériel clé actuel est utilisé à la fois pour le chiffrement et le déchiffrement, mais les autres matériaux clés (non courants) ne peuvent être utilisés que pour le déchiffrement.

Vous ne pouvez pas modifier l’origine des éléments de clé

Les clés KMS conçues pour les éléments importés sont dotées d'une valeur origin (origine) non modifiable définie sur `EXTERNAL`. Vous ne pouvez pas convertir une clé KMS pour du matériel

clé importé afin d'utiliser du matériel clé provenant d'une autre source, y compris AWS KMS. De même, vous ne pouvez pas convertir une clé KMS AWS KMS contenant du matériel clé en une clé conçue pour le matériel clé importé.

Vous ne pouvez pas exporter d’élément de clé

Vous ne pouvez pas exporter de matériel clé que vous avez importé. AWS KMS ne peut pas vous renvoyer le matériel clé importé sous quelque forme que ce soit. Vous devez conserver une copie du matériel clé importé à l'extérieur AWS, de préférence dans un gestionnaire de clés, tel qu'un module de sécurité matériel (HSM), afin de pouvoir réimporter le matériel clé si vous le supprimez ou s'il expire.

Vous pouvez créer des clés multi-région avec des éléments de clé importés

Les zones multirégionales avec un élément de clé importé ont les fonctionnalités des clés KMS avec un élément de clé importé et peuvent interagir entre Régions AWS. Pour créer une clé multi-région avec des éléments de clé importés, vous devez importer les mêmes éléments de clé dans la clé KMS principale et dans chaque clé de réplica. Les clés de chiffrement symétriques multirégionales ne prennent pas en charge la rotation à la demande.

Les clés asymétriques et les clés HMAC sont portables et interopérables

Vous pouvez utiliser le matériau de votre clé asymétrique et le matériau de votre clé HMAC à l'extérieur AWS pour interagir avec des AWS KMS clés contenant le même matériau de clé importé.

Contrairement au texte chiffré AWS KMS symétrique, qui est inextricablement lié à la clé KMS utilisée dans l'algorithme, AWS KMS utilise des formats HMAC standard et asymétriques pour le chiffrement, la signature et la génération de MAC. Par conséquent, les clés sont portables et prennent en charge les scénarios de clé sous séquestre traditionnels.

Lorsque votre clé KMS contient du matériel clé importé, vous pouvez utiliser le matériel clé importé AWS à l'extérieur pour effectuer les opérations suivantes.

- Clés HMAC – Vous pouvez vérifier une balise HMAC générée par la clé KMS HMAC avec un élément de clé importé. Vous pouvez également utiliser la clé HMAC KMS avec le matériel clé importé pour vérifier une étiquette HMAC qui a été générée par le matériel clé à l'extérieur de. AWS
- Clés de chiffrement asymétriques — Vous pouvez utiliser votre clé de chiffrement asymétrique privée AWS à l'extérieur pour déchiffrer un texte chiffré par la clé KMS avec la clé publique correspondante. Vous pouvez également utiliser votre clé KMS asymétrique pour déchiffrer un texte chiffré asymétrique généré en dehors de. AWS

- Clés de signature asymétriques — Vous pouvez utiliser votre clé KMS de signature asymétrique avec du matériel clé importé pour vérifier les signatures numériques générées par votre clé de signature privée en dehors de. AWS Vous pouvez également utiliser votre clé de signature publique asymétrique AWS à l'extérieur pour vérifier les signatures générées par votre clé KMS asymétrique.
- Clés d'accord de clé asymétriques — Vous pouvez utiliser votre clé KMS d'accord de clé asymétrique avec du matériel clé importé pour obtenir des secrets partagés avec un homologue extérieur à. AWS

Si vous importez les mêmes éléments de clé dans différentes clés KMS du même Région AWS, ces clés sont également interopérables. Pour créer des clés KMS interopérables dans différentes régions Régions AWS, créez une clé multirégionale avec du matériel clé importé.

Les clés de chiffrement symétriques ne sont ni portables ni interopérables

Les textes chiffrés symétriques AWS KMS produits ne sont ni portables ni interopérables. AWS KMS ne publie pas le format de texte chiffré symétrique requis par la portabilité, et le format peut changer sans préavis.

- AWS KMS ne peut pas déchiffrer les textes chiffrés symétriques que vous chiffrez en dehors de ceux-ci AWS, même si vous utilisez des éléments clés que vous avez importés.
- AWS KMS ne prend pas en charge le déchiffrement d'un texte chiffré AWS KMS symétrique en dehors de AWS KMS, même si le texte chiffré a été chiffré sous une clé KMS avec du matériel clé importé.
- Les clés KMS avec le même élément de clé importé ne sont pas interopérables. Le texte chiffré symétrique qui AWS KMS génère un texte chiffré spécifique à chaque clé KMS. Ce format de texte chiffré garantit que seule la clé KMS qui a chiffré des données peut les déchiffrer.

En outre, vous ne pouvez utiliser aucun AWS outil, tel que le AWS Encryption SDKchiffrement côté client Amazon S3, pour déchiffrer AWS KMS des textes chiffrés symétriques.

Par conséquent, vous ne pouvez pas utiliser de clés contenant du matériel clé importé pour soutenir des accords d'entiercement de clés dans le cadre desquels un tiers autorisé ayant un accès conditionnel au contenu clé peut déchiffrer certains textes chiffrés en dehors de. AWS KMS Pour prendre en charge le séquestre de clés, utilisez le kit AWS Encryption SDK pour chiffrer votre message sous une clé indépendante de AWS KMS.

Suppression des éléments de clé importés

Les éléments de clé que vous importez sont protégés pendant le transport et au repos. Avant d'importer le matériel clé, vous chiffrez (ou « enveloppez ») le contenu clé avec la clé publique d'une paire de clés RSA générée dans des modules de sécurité AWS KMS matériels (HSMs) validés dans le cadre du programme de validation des modules cryptographiques FIPS 140-3. Vous pouvez chiffrer l'élément de clé directement avec la clé publique d'enveloppement, ou chiffrer l'élément de clé avec une clé symétrique AES, puis chiffrer la clé symétrique AES avec la clé publique RSA.

À réception, AWS KMS déchiffre le contenu de la clé avec la clé privée correspondante dans un AWS KMS HSM et le chiffre à nouveau sous une clé symétrique AES qui n'existe que dans la mémoire volatile du HSM. Votre élément de clé ne quitte jamais le HSM en texte brut. Il est déchiffré uniquement lorsqu'il est utilisé et uniquement en cours d'utilisation. AWS KMS HSMs

L'utilisation de votre clé KMS avec l'élément de clé importé est déterminée uniquement par les politiques de contrôle d'accès que vous définissez sur la clé KMS. En outre, vous pouvez utiliser des alias et des balises pour identifier et contrôler l'accès à la clé KMS. Vous pouvez activer et désactiver la clé, la visualiser et la surveiller à l'aide de services tels que AWS CloudTrail.

Cependant, vous conservez la seule copie sûre de votre élément de clé. En échange de cette mesure de contrôle supplémentaire, vous êtes responsable de la durabilité et de la disponibilité globale du matériau clé importé. AWS KMS est conçu pour maintenir la haute disponibilité du matériel clé importé. Mais AWS KMS ne maintient pas la durabilité du matériau clé importé au même niveau que le matériau clé qui en AWS KMS génère.

Cette différence en durabilité est significative dans les cas suivants :

- Lorsque vous définissez une date d'expiration pour votre matériel clé importé, le AWS KMS matériel clé est supprimé après son expiration. AWS KMS ne supprime pas la clé KMS ni ses métadonnées. Vous pouvez créer une CloudWatch alarme Amazon qui vous avertit lorsque le matériel clé importé approche de sa date d'expiration.

  Vous ne pouvez pas supprimer le contenu clé AWS KMS généré pour une clé KMS et vous ne pouvez pas configurer le contenu AWS KMS clé pour qu'il expire.
- Lorsque vous supprimez manuellement le contenu clé importé, il AWS KMS supprime le contenu clé mais ne supprime pas la clé KMS ni ses métadonnées. En revanche, la planification de la suppression des clés nécessite une période d'attente de 7 à 30 jours, après quoi la clé KMS, ses métadonnées et son contenu clé sont AWS KMS définitivement supprimés.

- Dans le cas peu probable de certaines défaillances régionales susceptibles de l'affecter AWS KMS (comme une perte totale de courant), vous AWS KMS ne pourrez pas restaurer automatiquement le matériel clé importé. Cependant, AWS KMS vous pouvez restaurer la clé KMS et ses métadonnées.

Vous devez conserver une copie du matériel clé importé à l'extérieur d' AWS un système que vous contrôlez. Nous vous recommandons de stocker une copie exportable des éléments de clé importés dans un système de gestion des clés, tel qu'un module de sécurité matérielle (HSM). Il est recommandé de stocker une référence à l'ARN de la clé KMS et à l'ID du matériau clé généré par le AWS KMS biais de la copie exportable du contenu clé. Si l'élément de clé importé est supprimé ou expire, la clé KMS associée devient inutilisable tant que vous ne le réimportez pas. En cas de perte définitive des éléments de clé importés, tout texte chiffré au moyen de la clé KMS est irrécupérable.

> **Important**
>
> Les clés de chiffrement symétriques à région unique peuvent être associées à plusieurs éléments clés. La clé KMS dans son intégralité devient inutilisable dès que vous supprimez l'un de ces éléments clés ou si l'un de ces éléments clés expire (sauf si le matériel clé supprimé ou expirant le soit`PENDING_ROTATION`). Vous devez réimporter tous les éléments clés expirés ou supprimés associés à une telle clé avant que celle-ci ne soit utilisable pour des opérations cryptographiques.

Clés KMS dans un magasin de clés CloudHSM

Vous pouvez créer, afficher, gérer, utiliser et planifier la suppression du AWS KMS keys dans un magasin de AWS CloudHSM clés. Les procédures que vous employez sont très similaires à celles que vous utilisez pour les autres clés KMS. La seule différence est que vous spécifiez un magasin de AWS CloudHSM clés lorsque vous créez la clé KMS. AWS KMS Crée ensuite du matériel clé non extractible pour la clé KMS dans le AWS CloudHSM cluster associé au magasin de AWS CloudHSM clés. Lorsque vous utilisez une clé KMS dans un magasin de AWS CloudHSM clés, les opérations cryptographiques sont effectuées HSMs dans le cluster.

Fonctionnalités prises en charge

Outre les procédures décrites dans cette section, vous pouvez effectuer les opérations suivantes avec les clés KMS dans un magasin de AWS CloudHSM clés :

- Utiliser les politiques de clé, les politiques IAM et les octrois pour autoriser l'accès aux clés KMS.
- Activer et désactiver les clés KMS.
- Attribuer des balises, créer des alias et utiliser le contrôle d'accès par attributs (ABAC) pour autoriser l'accès aux clés KMS.
- Utilisez les clés KMS pour effectuer les opérations cryptographiques suivantes :
  - Encrypt
  - Decrypt
  - GenerateDataKey
  - GenerateDataKeyWithoutPlaintext
  - ReEncrypt

  Les opérations qui génèrent des paires de clés de données asymétriques GenerateDataKeyPairet GenerateDataKeyPairWithoutPlaintextne sont pas prises en charge dans les magasins de clés personnalisés.
- Utiliser les clés KMS avec les services AWS qui s'intègrent à AWS KMS et prennent en charge les clés gérées par le client.
- Suivez l'utilisation de vos clés KMS dans les AWS CloudTrail journaux et les outils CloudWatch de surveillance Amazon.

Fonctions non prises en charge

- AWS CloudHSM les magasins de clés ne prennent en charge que le chiffrement symétrique des clés KMS. Vous ne pouvez pas créer de clés HMAC KMS, de clés KMS asymétriques ou de paires de clés de données asymétriques dans un AWS CloudHSM magasin de clés.
- Vous ne pouvez pas importer de matériel clé dans une clé KMS dans un magasin de AWS CloudHSM clés. AWS KMS génère le matériel clé pour la clé KMS dans le AWS CloudHSM cluster.
- Vous ne pouvez pas activer ou désactiver la rotation automatique du contenu clé d'une clé KMS dans un magasin de AWS CloudHSM clés.

Utilisation de clés KMS dans un magasin de AWS CloudHSM clés

Lorsque vous utilisez votre clé KMS dans une demande, identifiez-la par son ID ou son alias ; il n'est pas nécessaire de spécifier le magasin de AWS CloudHSM clés ou le AWS CloudHSM cluster. La réponse inclut les mêmes champs qui sont renvoyés pour une clé KMS de chiffrement symétrique.

Toutefois, lorsque vous utilisez une clé KMS dans un magasin de AWS CloudHSM clés, l'opération cryptographique est entièrement réalisée au sein du AWS CloudHSM cluster associé au magasin de AWS CloudHSM clés. L'opération utilise les éléments de clé du cluster associés à la clé KMS que vous avez choisie.

Pour que cela soit possible, les conditions suivantes sont requises.

- L'état de la clé KMS doit être `Enabled`. Pour trouver l'état clé, utilisez le champ Status de la AWS KMS console ou le `KeyState` champ de la DescribeKeyréponse.
- Le magasin de AWS CloudHSM clés doit être connecté à son AWS CloudHSM cluster. Son statut dans la AWS KMS console ou `ConnectionState` dans la DescribeCustomKeyStoresréponse doit être`CONNECTED`.
- Le AWS CloudHSM cluster associé au magasin de clés personnalisé doit contenir au moins un HSM actif. Pour connaître le nombre de personnes actives HSMs dans le cluster, utilisez la AWS KMS console, la AWS CloudHSM console ou l'DescribeClustersopération.
- Le AWS CloudHSM cluster doit contenir le matériel clé pour la clé KMS. Si la clé a été supprimé du cluster, ou qu'un module HSM a été créé à partir d'une sauvegarde qui n'inclut pas la clé de chiffrement, l'opération de chiffrement échoue.

Si ces conditions ne sont pas remplies, l'opération cryptographique échoue et AWS KMS renvoie une `KMSInvalidStateException` exception. En général, il suffit de reconnecter le magasin de AWS CloudHSM clés. Pour obtenir de l'aide supplémentaire, consultez Comment corriger les clés KMS défaillantes.

Lorsque vous utilisez les clés KMS dans un magasin de AWS CloudHSM clés, sachez que les clés KMS de chaque AWS CloudHSM magasin de clés partagent un quota de demandes de stockage de clés personnalisé pour les opérations cryptographiques. Si vous dépassez le quota, AWS KMS renvoie un`ThrottlingException`. Si le AWS CloudHSM cluster associé au magasin de AWS CloudHSM clés traite de nombreuses commandes, y compris celles qui ne sont pas liées `ThrottlingException` au magasin de AWS CloudHSM clés, vous pouvez obtenir un taux encore plus faible. Si vous obtenez une exception `ThrottlingException` pour une demande, réduisez la fréquence des demandes et essayez les commandes à nouveau. Pour plus d'informations sur le quota de magasin de clés personnalisé, veuillez consulter la rubrique Quotas de demandes de magasin de clés personnalisé.

En savoir plus

- Pour en savoir plus sur les magasins à AWS CloudHSM clés, consultezAWS CloudHSM magasins clés.

- Pour créer des clés KMS dans un magasin de AWS CloudHSM clés, consultezCréation d'une clé KMS dans un magasin de AWS CloudHSM clés.
- Pour identifier et afficher les clés KMS dans un magasin de AWS CloudHSM clés, voirIdentifiez les clés KMS dans les magasins de AWS CloudHSM clés.
- Pour trouver des clés KMS et du matériel clé dans un magasin de AWS CloudHSM clés, voirTrouvez les clés KMS et le matériel clé dans un magasin de AWS CloudHSM clés.
- Pour en savoir plus sur les considérations particulières relatives à la suppression de clés KMS dans un magasin de AWS CloudHSM clés, consultez la section Suppression de clés KMS d'un magasin de AWS CloudHSM clés.

Clés KMS dans des magasins de clés externes

Pour créer, afficher, gérer, utiliser et planifier la suppression des clés KMS d'un magasin de clés externe, vous devez utiliser des procédures très similaires à celles que vous utilisez pour les autres clés KMS. Toutefois, lorsque vous créez une clé KMS dans un magasin de clés externe, vous spécifiez un magasin de clés externe et une clé externe. Lorsque vous utilisez une clé KMS dans un magasin de clés externe, les opérations de chiffrement et de déchiffrement sont effectuées par votre gestionnaire de clés externe à l'aide de la clé externe spécifiée.

AWS KMS Impossible de créer, d'afficher, de mettre à jour ou de supprimer des clés cryptographiques dans votre gestionnaire de clés externe. AWS KMS n'accède jamais directement à votre gestionnaire de clés externe ni à aucune clé externe. Toutes les requêtes d'opérations cryptographiques sont acheminées par votre proxy de magasin de clés externe. Pour utiliser une clé KMS dans un magasin de clés externe, le magasin de clés externe qui héberge la clé KMS doit être connecté à son proxy de magasin de clés externe.

Fonctionnalités prises en charge

Outre les procédures décrites dans cette section, vous pouvez effectuer les actions suivantes avec les clés KMS d'un magasin de clés externe :

- Utiliser les politiques de clé, les politiques IAM et les octrois pour contrôler l'accès aux clés KMS.
- Activer et désactiver les clés KMS. Ces actions n'affectent pas la clé externe dans votre gestionnaire de clés externe.
- Attribuez des balises, créez des alias et utilisez le contrôle d'accès par attributs (ABAC) pour autoriser l'accès aux clés KMS.

- Utilisez les clés KMS pour effectuer les opérations cryptographiques suivantes :
  - Encrypt
  - Decrypt
  - GenerateDataKey
  - GenerateDataKeyWithoutPlaintext
  - ReEncrypt

  Les opérations qui génèrent des paires de clés de données asymétriques GenerateDataKeyPairet GenerateDataKeyPairWithoutPlaintextne sont pas prises en charge dans les magasins de clés personnalisés.
- Utilisez les clés KMS avec les Services AWS qui s'intègrent à AWS KMS et prenez en charge les clés gérées par le client.

Fonctions non prises en charge

- Les magasins de clés externes ne prennent en charge que les clés KMS de chiffrement symétriques. Vous ne pouvez pas créer de clés KMS HMAC ou de clés KMS asymétriques dans un magasin de clés externe.
- GenerateDataKeyPairet ne GenerateDataKeyPairWithoutPlaintextsont pas pris en charge sur les clés KMS d'un magasin de clés externe.
- Vous ne pouvez pas utiliser un AWS::KMS::Key AWS CloudFormation modèle pour créer un magasin de clés externe ou une clé KMS dans un magasin de clés externe.
- Les clés multi-régions ne sont pas prises en charge dans un magasin de clés externe.
- Les clés KMS contenant des éléments de clé importés ne sont pas prises en charge dans un magasin de clés externe.
- La rotation automatique des clés n'est pas prise en charge pour les clés KMS dans un magasin de clés externe.

Utilisation de clés KMS dans un magasin de clés externe

Lorsque vous utilisez votre clé KMS dans une requête, identifiez la clé KMS par son ID de clé, son ARN de clé, son alias ou son ARN d'alias. Vous n'avez pas besoin de spécifier le magasin de clés externe. La réponse inclut les mêmes champs qui sont renvoyés pour une clé KMS de chiffrement symétrique. Toutefois, lorsque vous utilisez une clé KMS dans un magasin de clés externe, les opérations de chiffrement et de déchiffrement sont effectuées par votre gestionnaire de clés externe au moyen de la clé externe associée à la clé KMS.

Pour garantir que le texte chiffré par une clé KMS dans un magasin de clés externe est au moins aussi sûr que tout texte chiffré par une clé KMS standard, AWS KMS utilisez le double chiffrement. Les données sont d'abord cryptées à AWS KMS l'aide de matériel AWS KMS clé. Elles sont ensuite chiffrées par votre gestionnaire de clés externe à l'aide de la clé externe de la clé KMS. Pour déchiffrer un texte chiffré à double chiffrement, le texte chiffré est d'abord déchiffré par votre gestionnaire de clés externe à l'aide de la clé externe de la clé KMS. Ensuite, il est déchiffré en AWS KMS utilisant le matériau AWS KMS clé de la clé KMS.

Pour que cela soit possible, les conditions suivantes sont requises.

- L'état de la clé KMS doit être `Enabled`. Pour connaître l'état de la clé, consultez le champ État pour les clés gérées par le client, sur la AWS KMS console ou dans le `KeyState` champ de la DescribeKeyréponse.
- Le magasin de clés externe qui héberge la clé KMS doit être connecté à son proxy de magasin de clés externe, c'est-à-dire que l'état de connexion du magasin de clés externe doit être `CONNECTED`.

  Vous pouvez consulter l'état de la connexion sur la page Stockages de clés externes de la AWS KMS console ou dans la DescribeCustomKeyStoresréponse. L'état de connexion du magasin de clés externe est également affiché sur la page de détails de la clé KMS sur la console AWS KMS . Sur la page de détails, choisissez l'onglet Cryptographic configuration (Configuration cryptographique) et consultez le champ Connection state (État de la connexion) dans la section Custom key store (Magasin de clés personnalisé).

  Si l'état de connexion est `DISCONNECTED`, vous devez d'abord le connecter. Si l'état de connexion est `FAILED`, vous devez résoudre le problème, déconnecter le magasin de clés externe, puis le connecter. Pour obtenir des instructions, consultez Connecter et déconnecter les magasins de clés externes.
- Le proxy de magasin de clés externe doit être en mesure de trouver la clé externe.
- La clé externe doit être activée et elle doit effectuer le chiffrement et le déchiffrement.

  L'état de la clé externe est indépendant et n'est pas affecté par les modifications de l'état de clé de la clé KMS, y compris par l'activation et la désactivation de la clé KMS. De même, la désactivation ou la suppression de la clé externe ne modifie pas l'état de la clé KMS, mais les opérations cryptographiques utilisant la clé KMS associée échoueront.

Si ces conditions ne sont pas remplies, l'opération cryptographique échoue et AWS KMS renvoie une `KMSInvalidStateException` exception. Vous devrez peut-être reconnecter le magasin

de clés externe ou utiliser les outils de votre gestionnaire de clés externe pour reconfigurer ou réparer votre clé externe. Pour obtenir de l'aide supplémentaire, consultez the section called “Résoudre les problèmes liés aux magasins de clés externes”.

Lorsque vous utilisez les clés KMS dans un magasin de clés externe, sachez que les clés KMS de chaque magasin de clés externe partagent un quota de magasin de clés personnalisé sur les requêtes d'opérations cryptographiques. Si vous dépassez le quota, AWS KMS renvoie un`ThrottlingException`. Pour plus d'informations sur le quota de magasin de clés personnalisé, veuillez consulter la rubrique Quotas de demandes de magasin de clés personnalisé.

En savoir plus

- Pour en savoir plus sur les magasins de clés externes, consultezMagasins de clés externes.
- Pour en savoir plus sur les informations clés contenues dans les magasins de clés externes, consultezClé externe.
- Pour créer des clés KMS dans un magasin de clés externe, consultezCréation d'une clé KMS dans des magasins de clés externes.
- Pour identifier et afficher les clés KMS dans un magasin de clés externe, voirIdentifier les clés KMS dans les magasins de clés externes.
- Pour en savoir plus sur les considérations particulières relatives à la suppression de clés KMS dans un magasin de clés externe, voir Suppression de clés KMS d'un magasin de clés externe.

AWS KMS éléments essentiels de la cryptographie

AWS KMS utilise des algorithmes cryptographiques configurables afin que le système puisse rapidement passer d'un algorithme ou d'un mode approuvé à un autre. L'ensemble initial d'algorithmes cryptographiques par défaut a été sélectionné dans les algorithmes Federal Information Processing Standard (FIPS Approved) pour leurs propriétés de sécurité et leurs performances.

Entropie et génération de nombres aléatoires

AWS KMS la génération de clés est effectuée dans le AWS KMS HSMs. Ils HSMs implémentent un générateur de nombres aléatoires hybride qui utilise le NIST SP800-90A Deterministic Random Bit Generator (DRBG) CTR_DRBG using AES-256. Il est alimenté par un générateur de bits aléatoires non déterministe avec 384 bits d'entropie et mis à jour avec de l'entropie supplémentaire aux fins de fournir une résistance à la prédiction à chaque appel d'élément cryptographique.

Opérations de clé symétrique (chiffrement uniquement)

Toutes les commandes de chiffrement par clé symétrique utilisées dans ce cadre HSMs utilisent les normes de chiffrement avancées (AES), en mode compteur Galois (GCM) à l'aide de clés de 256 bits. Les appels analogues pour déchiffrer utilisent la fonction inverse.

AES-GCM est un schéma de chiffrement authentifié. En plus de chiffrer le texte brut afin de produire du texte chiffré, il calcule une balise d'authentification sur le texte chiffré et toutes les données supplémentaires pour lesquelles une authentification est requise (données authentifiées supplémentaires, ou AAD). La balise d'authentification permet de s'assurer que les données proviennent de la source présumée et que le texte chiffré et l'AAD n'ont pas été modifiés.

AWS Omet souvent l'inclusion de l'AAD dans nos descriptions, en particulier lorsqu'il s'agit du chiffrement des clés de données. Dans ces cas, le texte environnant laisse entendre que la structure à chiffrer est divisée entre le texte brut à chiffrer et l'AAD en texte clair à protéger

AWS KMS vous offre la possibilité d'importer du matériel clé dans un AWS KMS key au lieu de compter sur AWS KMS celui-ci pour générer le matériel clé. Ce matériel clé importé peut être chiffré à l'aide du protocole RSAES-OAEP pour protéger la clé pendant le transport vers le HSM. AWS KMS Les paires de clés RSA sont générées sur AWS KMS HSMs. Le matériel clé importé est déchiffré sur un AWS KMS HSM et rechiffré sous AES-GCM avant d'être stocké par le service.

Opérations de clés asymétriques (chiffrement, signature numérique et vérification de signature)

AWS KMS prend en charge l'utilisation d'opérations de clé asymétriques pour les opérations de chiffrement, de signature numérique et d'accord de clés. Les opérations de clé asymétrique reposent sur une paire de clés publiques et de clés privées liées mathématiquement que vous pouvez utiliser pour le chiffrement et le déchiffrement, la signature et la vérification des signatures, ou pour obtenir des secrets partagés. La clé privée ne sort jamais AWS KMS non chiffrée. Vous pouvez utiliser la clé publique interne AWS KMS en appelant les opérations de l' AWS KMS API, ou télécharger la clé publique et l'utiliser en dehors de AWS KMS.

AWS KMS prend en charge les chiffrements asymétriques suivants.

- RSA-OAEP (pour le chiffrement) et RSA-PSS et RSA-PKCS- #1 -v1_5 (pour la signature et la vérification) – Prend en charge les longueurs de clés RSA (en bits) : 2 048, 3 072 et 4 096 pour différentes exigences de sécurité.

- Courbe elliptique (ECC) : utilisée pour signer et vérifier ou pour obtenir des secrets partagés, mais pas les deux. Prend en charge les courbes ECC : NIST P256, P384, P521, SECP 256k1.
- ML-DSA — Utilisé pour la signature et la vérification. Les principales spécifications ML-DSA prises en charge sont les suivantes : ML_DSA_44, ML_DSA_65 et ML_DSA_87.
- SM2 (Régions de Chine uniquement) — Utilisé pour le chiffrement et le déchiffrement, la signature et la vérification, ou pour la dérivation de secrets partagés, mais vous devez choisir une seule utilisation de clé. Supporte le SM2 protocole PKE pour le chiffrement et le protocole SM2 DSA pour la signature.

Fonctions de dérivation de clé

Une fonction de dérivation de clés est utilisée pour dériver des clés supplémentaires à partir d'un secret ou d'une clé initiale. AWS KMS utilise une fonction de dérivation de clés (KDF) pour dériver des clés par appel pour chaque chiffrement effectué sous un. AWS KMS key Toutes les opérations KDF utilisent le KDF en mode compteur en utilisant HMAC [FIPS197] avec SHA256 [FIPS180]. La clé dérivée de 256 bits est utilisée avec AES-GCM aux fins de chiffrer ou de déchiffrer les données et les clés des clients.

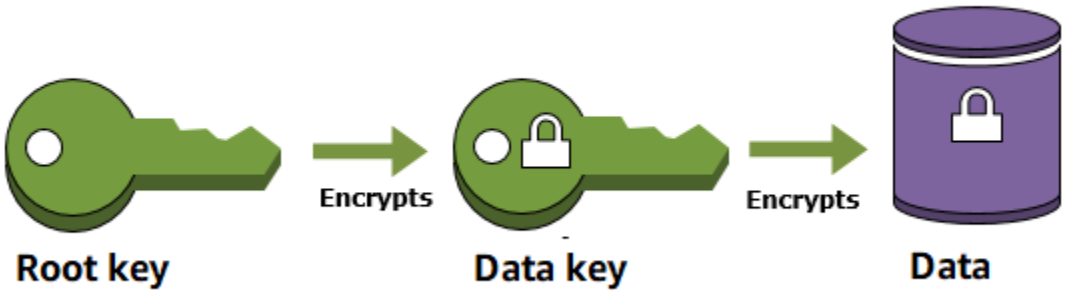
AWS KMS utilisation interne de signatures numériques

Les signatures numériques sont également utilisées pour authentifier des commandes et des communications entre AWS KMS entités. Toutes les entités de service disposent d'une paire de clés de l'algorithme de signature numérique à courbe elliptique (ECDSA). Elles exécutent l'ECDSA comme décrit dans Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS) and X9.62-2005: Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA). Les entités utilisent l'algorithme de hachage sécurisé défini dans les publications sur les normes fédérales de traitement de l'information, FIPS PUB 180-4, connu sous le nom de. SHA384 Les clés sont générées sur la courbe secp384r1 (NIST-P384).

Chiffrement d’enveloppe

Lorsque vous chiffrez vos données, celles-ci sont protégées, mais vous devez protéger votre clé de chiffrement. Une stratégie consiste à la chiffrer. Le chiffrement d'enveloppe est la pratique consistant à chiffrer des données en texte brut à l'aide d'une clé de données, puis à chiffrer la clé de données sous une autre clé.

Vous pouvez même chiffrer la clé de chiffrement de données sous une autre clé de chiffrement et chiffrer cette clé de chiffrement sous une autre clé de chiffrement. Toutefois, au final, une clé doit rester en texte brut pour vous permettre de déchiffrer les clés et vos données. Cette clé de chiffrement de clé en texte brut de niveau supérieur porte le nom de clé racine.



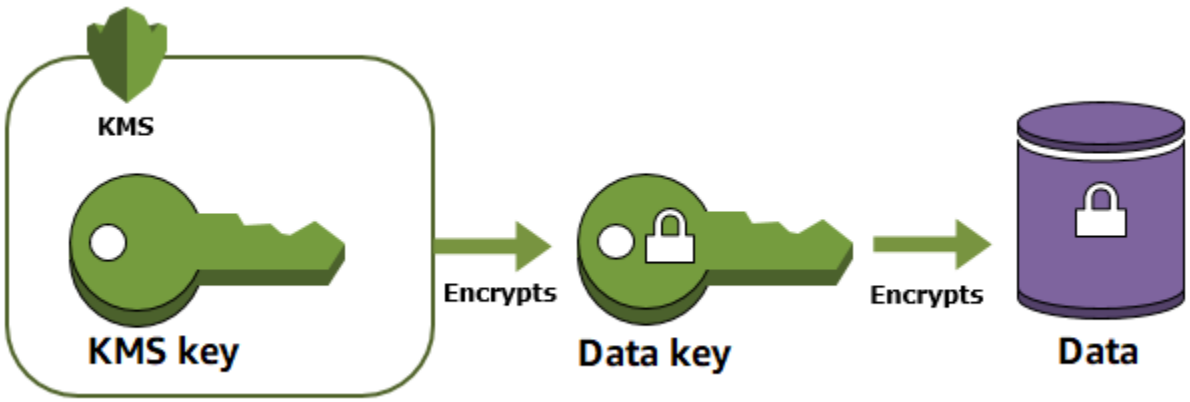
AWS KMS vous aide à protéger vos clés de chiffrement en les stockant et en les gérant en toute sécurité. La clé racine stockée dans AWS KMS, connue sous le nom de AWS KMS keys, ne laissez jamais les modules de sécurité matériels validés par la norme de sécurité AWS KMS FIPS 140-3 de niveau 3 non chiffrés. Pour utiliser une clé KMS, vous devez appeler AWS KMS.

Une construction de base utilisée dans de nombreux systèmes cryptographiques est le chiffrement d'enveloppe. Le chiffrement d'enveloppe utilise deux clés cryptographiques ou plus afin de sécuriser un message. Généralement, une clé est dérivée d'une clé statique à plus long terme k, et une autre clé est une clé par message, msgKey, qui est générée pour chiffrer le message. L'enveloppe est constituée en chiffrant le message :texte chiffré = Encrypt(msgKey, message). Ensuite, la clé de message est chiffrée à l'aide de la clé statique à long terme : encKey = Encrypt(k, msgKey). Enfin, les deux valeurs (encKey, texte chiffré) sont empaquetés dans une structure unique, ou dans un message chiffré par enveloppe.

Le destinataire, avec accès à k, peut ouvrir le message enveloppé en déchiffrant d'abord la clé chiffrée, puis en déchiffrant le message.

AWS KMS permet de gérer ces clés statiques à long terme et d'automatiser le processus de chiffrement des enveloppes de vos données.

Outre les fonctionnalités de chiffrement fournies par le AWS KMS service, le SDK de AWS chiffrement fournit des bibliothèques de chiffrement d'enveloppes côté client. Vous pouvez utiliser ces bibliothèques pour protéger vos données et les clés de chiffrement utilisées pour chiffrer ces données.



Le chiffrement d'enveloppe offre plusieurs avantages :

- Protection des clés de données

  Lorsque vous chiffrez une clé de données, vous n'avez pas à vous préoccuper du stockage de la clé de données chiffrée, car cette clé de données est intrinsèquement protégée par chiffrement. Vous pouvez stocker en toute sécurité la clé de données chiffrée avec les données chiffrées.

- Chiffrement des mêmes données sous plusieurs clés

  Les opérations de chiffrement peuvent exiger beaucoup de temps, notamment lorsque les données en cours de chiffrement sont des objets de grande taille. Au lieu de rechiffrer des données brutes plusieurs fois avec des clés différentes, vous pouvez rechiffrer uniquement les clés de données qui protègent les données brutes.

- Combinaison des points forts de plusieurs algorithmes

  En général, les algorithmes de clé symétrique sont plus rapides et produisent des textes chiffrés plus petits que les algorithmes de clé publique. Cependant, les algorithmes de clé publique fournissent une séparation inhérente des rôles et facilitent la gestion des clés. Le chiffrement d'enveloppe vous permet d'associer les forces de chaque stratégie.

Opérations cryptographiques

Dans AWS KMS, les opérations cryptographiques sont des opérations d'API qui utilisent des clés KMS pour protéger les données. Comme les clés KMS restent à l'intérieur AWS KMS, vous devez appeler AWS KMS pour utiliser une clé KMS dans le cadre d'une opération cryptographique.

Pour effectuer des opérations cryptographiques avec des clés KMS, utilisez le AWS SDKs, AWS Command Line Interface (AWS CLI) ou le Outils AWS pour PowerShell. Vous ne pouvez pas effectuer d'opérations cryptographiques dans la console AWS KMS . Pour obtenir des exemples d'appel des opérations cryptographiques dans plusieurs langages de programmation, veuillez consulter Exemples de code pour AWS KMS l'utilisation AWS SDKs.

Le tableau suivant répertorie les opérations AWS KMS cryptographiques. Il indique également le type de clé et les exigences d'utilisation des clés KMS utilisées dans l'opération.

Opération	Type de clé	Utilisation de la clé
Decrypt	Symétrique ou asymétrique	`ENCRYPT_DECRYPT`
DeriveSharedSecret	Asymétrique	`KEY_AGREEMENT`
Encrypt	Symétrique ou asymétrique	`ENCRYPT_DECRYPT`
GenerateDataKey	Symétrique	`ENCRYPT_DECRYPT`
GenerateDataKeyPair	Symétrique [1] Non pris en charge sur les clés KMS dans les magasins de clés personnalisés.	`ENCRYPT_DECRYPT`
GenerateDataKeyPairWithoutPlaintext	Symétrique [1] Non pris en charge sur les clés KMS dans les magasins de clés personnalisés.	`ENCRYPT_DECRYPT`
GenerateDataKeyWithoutPlaintext	Symétrique	`ENCRYPT_DECRYPT`
GenerateMac	HMAC	`GENERATE_VERIFY_MAC`

Opération	Type de clé	Utilisation de la clé
GenerateRandom	N/A. Cette opération n'utilise pas de clé KMS.	N/A
ReEncrypt	Symétrique ou asymétrique	`ENCRYPT_DECRYPT`
Sign (Signer)	Asymétrique	`SIGN_VERIFY`
Vérification	Asymétrique	`SIGN_VERIFY`
VerifyMac	HMAC	`GENERATE_VERIFY_MAC`

[1] Génère une paire de clés de données asymétriques qui est protégée par une clé KMS de chiffrement symétrique.

Pour plus d'informations sur les autorisations pour les opérations cryptographiques, veuillez consulter the section called “Référence des autorisations ”.

Pour que tous les utilisateurs soient AWS KMS réactifs et hautement fonctionnels, AWS KMS établissez des quotas sur le nombre d'opérations cryptographiques appelées par seconde. Pour en savoir plus, consultez the section called “Quotas partagés pour les opérations de chiffrement”.

Accès aux clés KMS et autorisations

Pour l'utiliser AWS KMS, vous devez disposer d'informations d'identification AWS permettant d'authentifier vos demandes. Les informations d'identification doivent inclure les autorisations d'accès aux AWS ressources : AWS KMS keys et les alias. Aucun AWS principal n'a d'autorisation sur une clé KMS à moins que cette autorisation ne soit fournie explicitement et ne soit jamais refusée. Il n'existe aucune autorisation implicite ou automatique pour utiliser ou gérer une clé KMS.

Pour contrôler l'accès à vos clés KMS, vous pouvez utiliser les mécanismes de politique suivants.

- Politique de clé – chaque clé KMS a une politique de clé. Il s'agit du principal mécanisme de contrôle d'accès à une clé KMS. Vous pouvez utiliser la politique de clé seule pour contrôler l'accès, ce qui signifie que l'étendue complète de l'accès à la clé KMS est définie dans un document unique (la politique de clé). Pour plus d'informations sur l'utilisation des politiques de clé, consultez la rubrique Politiques de clé.
- Politiques IAM – vous pouvez utiliser des politiques IAM en combinaison avec la politique de clé et les octrois pour contrôler l'accès à une clé KMS. Contrôler l'accès de cette façon vous permet de gérer toutes les autorisations pour vos identités IAM dans IAM. Pour utiliser une politique IAM pour autoriser l'accès à une clé KMS, la politique de clé doit l'autoriser explicitement. Pour en savoir plus sur l'utilisation de politiques IAM, veuillez consulter Politiques IAM.
- Octrois – Vous pouvez utiliser des octrois en association avec les politiques IAM et de clé pour autoriser l'accès à une clé KMS. En contrôlant l'accès de cette façon, vous pouvez autoriser l'accès à la clé KMS dans la politique de clé et permettre aux identités de déléguer leur accès à d'autres personnes. Pour plus d'informations sur l'utilisation des octrois, consultez la rubrique Subventions en AWS KMS.

Politiques clés de KMS

Les politiques constituent le principal moyen de gérer l'accès à vos AWS KMS ressources. Les politiques sont des documents qui décrivent quels principaux peuvent accéder à quelles ressources. Les politiques associées à une identité IAM sont appelées politiques basées sur l'identité (ou politiques IAM), et les politiques associées à d'autres types de ressources sont appelées politiques de ressources. AWS KMS les politiques de ressources pour les clés KMS sont appelées politiques clés.

Toutes les clés KMS ont une politique de clé. Si vous n'en fournissez pas, AWS KMS créez-en un pour vous. La politique de clé par défaut AWS KMS utilisée varie selon que vous créez la clé dans la AWS KMS console ou que vous utilisez l' AWS KMS API. Nous vous recommandons de modifier la politique clé par défaut afin de l'aligner sur les exigences de votre organisation en matière d'autorisations du moindre privilège.

Vous pouvez utiliser la politique des clés uniquement pour contrôler l'accès si la clé et le principal IAM se trouvent dans le même AWS compte, ce qui signifie que l'étendue complète de l'accès à la clé KMS est définie dans un seul document (la politique clé). Toutefois, lorsqu'un appelant d'un compte doit accéder à une clé d'un autre compte, vous ne pouvez pas utiliser uniquement la politique des clés pour accorder l'accès. Dans le scénario entre comptes, une politique IAM doit être associée à l'utilisateur ou au rôle de l'appelant pour autoriser explicitement l'appelant à effectuer l'appel d'API.

Vous pouvez également utiliser des politiques IAM en combinaison avec des politiques clés et des autorisations pour contrôler l'accès à une clé KMS. Pour utiliser une politique IAM afin de contrôler l'accès à une clé KMS, la politique de clé doit autoriser le compte à utiliser les politiques IAM. Vous pouvez soit spécifier une déclaration de politique clé qui active les politiques IAM, soit spécifier explicitement les principes autorisés dans la politique clé.

Lorsque vous rédigez des politiques, assurez-vous de disposer de contrôles stricts limitant les personnes autorisées à effectuer les actions suivantes :

- Mettre à jour, créer et supprimer les politiques clés IAM et KMS
- Associer et détacher les politiques IAM des utilisateurs, des rôles et des groupes
- Attachez et détachez les politiques relatives aux clés KMS à vos clés KMS

Subventions clés KMS

Outre l'IAM et les politiques clés, AWS KMS soutient les subventions. Les subventions constituent un moyen flexible et puissant de déléguer des autorisations. Vous pouvez utiliser des subventions pour délivrer un accès par clé KMS limité dans le temps aux principaux IAM de votre AWS compte ou d'autres comptes. AWS Nous vous recommandons de délivrer un accès limité dans le temps si vous ne connaissez pas le nom des principaux au moment de la création des politiques, ou si les principaux nécessitant un accès changent fréquemment. Le principal du bénéficiaire peut être sur le même compte que la clé KMS ou sur un compte différent. Si le principal et la clé KMS se trouvent dans des comptes différents, vous devez spécifier une politique IAM en plus de la subvention. Les

subventions nécessitent une gestion supplémentaire, car vous devez appeler une API pour créer la subvention et pour retirer ou révoquer l'autorisation lorsqu'elle n'est plus nécessaire.

Les rubriques suivantes fournissent des informations sur la manière dont vous pouvez utiliser AWS Identity and Access Management (IAM) et AWS KMS les autorisations pour sécuriser vos ressources en contrôlant les personnes autorisées à y accéder.

Politiques clés en AWS KMS

Une politique clé est une politique de ressources pour un AWS KMS key. Les politiques de clé constituent le principal moyen de contrôler l'accès aux clés KMS. Chaque clé KMS doit avoir exactement une politique de clé. Les instructions dans la politique de clé déterminent qui a l'autorisation d'utiliser la clé KMS et la façon dont ces personnes peuvent l'utiliser. Vous pouvez également utiliser les politiques IAM et les octrois pour contrôler l'accès à la clé KMS, mais chaque clé KMS doit avoir une politique de clé.

Aucun AWS principal, y compris l'utilisateur root du compte ou le créateur de la clé, n'est autorisé à accéder à une clé KMS, sauf si cela est explicitement autorisé, et jamais refusé, dans une politique clé, une politique IAM ou une subvention.

À moins que la politique de clé ne l'autorise explicitement, vous ne pouvez pas utiliser de politiques IAM pour autoriser l'accès à une clé KMS. Sans autorisation de la politique de clé, les politiques IAM qui octroient des autorisations n'ont aucun effet. (Vous pouvez utiliser une politique IAM pour refuser une autorisation à une clé KMS sans l'autorisation d'une politique de clé.) La politique de clé par défaut active les politiques IAM. Pour activer les politiques IAM dans votre politique de clé, ajoutez l'instruction de politique décrite dans Autorise l'accès au Compte AWS et active les politiques IAM..

Contrairement aux politiques IAM, qui sont mondiales, les politiques de clés sont régionales. Une politique de clé contrôle uniquement l'accès à une clé KMS dans la même région. Elle n'a aucun effet sur les clés KMS des autres régions.

Rubriques

- Création d'une politique de clé
- politique de clé par défaut
- Afficher une politique clé
- Modifier une politique clé

- Autorisations pour les AWS services dans les politiques clés

Création d'une politique de clé

Vous pouvez créer et gérer des politiques clés dans la AWS KMS console ou à l'aide d'opérations d' AWS KMS API CreateKey, telles que ReplicateKey, et PutKeyPolicy.

Lorsque vous créez une clé KMS dans la AWS KMS console, celle-ci vous explique les étapes de création d'une politique clé basée sur la politique clé par défaut de la console. Lorsque vous utilisez le `CreateKey` ou `ReplicateKey` APIs, si vous ne spécifiez pas de stratégie clé, ceux-ci APIs appliquent la politique clé par défaut pour les clés créées par programmation. Lorsque vous créez l'API `PutKeyPolicy`, vous devez spécifier une stratégie de clé.

Chaque document de politique peut contenir une ou plusieurs instruction(s) de politique. L'exemple suivant montre un document de politique de clé valide avec une instruction de politique.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Describe the policy statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:user/Alice"
      },
      "Action": "kms:DescribeKey",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:KeySpec": "SYMMETRIC_DEFAULT"
        }
      }
    }
  ]
}
```

Rubriques

- Format de politique de clé
- Éléments d'une politique de clé
- Exemple de politique de clé

Format de politique de clé

Un document de politique de clé doit être conforme aux règles suivantes :

- Jusqu'à 32 kilooctets (32 768 octets)
- L'élément `Sid` dans une instruction de politique de clé peut inclure des espaces. (Les espaces sont interdits dans l'élément `Sid` d'un document de politique IAM.)

Un document de politique de clé ne peut contenir que les caractères suivants :

- Caractères ASCII imprimables
- Caractères imprimables du jeu de caractères Basic Latin et du jeu de caractères supplémentaires Latin-1
- Les caractères spéciaux tabulation (`\u0009`), saut de ligne (`\u000A`) et retour chariot (`\u000D`)

Éléments d'une politique de clé

Un document de politique de clé doit disposer des éléments suivants :

Version

Spécifie la version du document de la politique de clé. Définissez la version sur `2012-10-17` (la dernière version).

Instruction

Comprend les instructions de la politique. Un document de politique de clé doit avoir au moins une instruction.

Chaque instruction de politique de clé est composée de six éléments. Les éléments `Effect`, `Principal`, `Action`, et `Resource` sont obligatoires.

Sid

(Facultatif) L'identifiant d'instruction (`Sid`) est une chaîne arbitraire que vous pouvez utiliser pour identifier l'instruction. Le `Sid` dans une politique de clé peut inclure des espaces. (Vous ne pouvez pas inclure d'espaces dans un élément `Sid` de politique IAM.)

Effet

(Requis) Détermine s'il convient d'autoriser ou de rejeter les autorisations figurant dans l'instruction de politique. Les valeurs valides sont `Allow` ou `Deny`. Si vous n'autorisez pas explicitement l'accès à une clé KMS, l'accès est implicitement refusé. Vous pouvez explicitement refuser l'accès à une clé KMS. Vous pouvez le faire afin de vous assurer qu'un utilisateur n'y a pas accès, même si une politique différente autorise l'accès.

Principal

(Requis) Le principal est l'identité qui obtient les autorisations figurant dans l'instruction de politique. Vous pouvez spécifier Comptes AWS des utilisateurs IAM, des rôles IAM et certains AWS services en tant que principaux dans une politique clé. Les groupes d'utilisateurs IAM ne constituent un principal valide dans aucun type de politique.

Une valeur astérisque, par exemple `"AWS": "*"`, représente toutes les identités AWS de tous les comptes.

> **Important**
>
> Ne définissez pas le principal sur un astérisque (*) dans une instruction de politique de clé qui autorise des autorisations, sauf si vous utilisez des conditions pour limiter la stratégie de clé. Un astérisque indique chaque identité associée à chaque Compte AWS autorisation d'utilisation de la clé KMS, sauf si une autre déclaration de politique le nie explicitement. Les utilisateurs des autres utilisateurs Comptes AWS peuvent utiliser votre clé KMS chaque fois qu'ils disposent des autorisations correspondantes sur leur propre compte.

> **Note**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Lorsque le principal dans une instruction de politique de clé est un Compte AWS principal exprimé comme `arn:aws:iam::111122223333:root"`, la déclaration de stratégie n'accorde aucune autorisation à un principal IAM. Il donne plutôt l' Compte AWS autorisation d'utiliser les politiques IAM pour déléguer les autorisations spécifiées dans la politique clé. (Un principal au format `arn:aws:iam::111122223333:root"` ne représente pas l'utilisateur racine du compte AWS, malgré l'utilisation de « root » [racine] dans l'identifiant du compte. Cependant, le principal du compte représente le compte et ses administrateurs, y compris l'utilisateur racine du compte.)

Lorsque le principal est un autre Compte AWS ou ses principaux, les autorisations ne sont effectives que lorsque le compte est activé dans la région avec la clé KMS et la politique de clé. Pour plus d'informations sur les régions qui ne sont pas activées par défaut (« Régions d'adhésion »), veuillez consulter Gestion de Régions AWS dans la Références générales AWS.

Pour autoriser un autre compte Compte AWS ou ses principaux utilisateurs à utiliser une clé KMS, vous devez fournir une autorisation dans une politique clé et dans une politique IAM de l'autre compte. Pour en savoir plus, consultez Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

Action

(Requis) Spécifiez les opérations d'API à autoriser ou à rejeter. Par exemple, l'`kms:Encrypt`action correspond à l'opération AWS KMS Chiffrer. Vous pouvez répertorier plusieurs actions dans une instruction de politique. Pour de plus amples informations, veuillez consulter Référence des autorisations .

> **Note**
>
> Si l'élément `Action` requis est absent d'une instruction de politique de clé, la déclaration de stratégie n'a aucun effet. Une déclaration de politique clé sans `Action` élément ne s'applique à aucune clé KMS.
> Lorsqu'il manque un `Action` élément à une déclaration de politique clé, la AWS KMS console signale correctement une erreur, mais le CreateKeyet PutKeyPolicy APIs réussit, même si la déclaration de politique est inefficace.

Ressource

(Requis) Dans une politique de clé, la valeur de l'élément Ressource est "*", ce qui signifie « cette clé KMS ». L'astérisque ("*") identifie la clé KMS à laquelle la politique de clé est attachée.

> **Note**
>
> Si l'élément `Resource` requis est absent d'une instruction de politique de clé, la déclaration de stratégie n'a aucun effet. Une instruction de politique de clé sans élément `Resource` ne s'applique à aucune clé KMS.
> Lorsqu'il manque un `Resource` élément à une déclaration de politique clé, la AWS KMS console signale correctement une erreur, mais le CreateKeyet PutKeyPolicy APIs réussit, même si la déclaration de politique est inefficace.

Condition

(Facultatif) Les conditions spécifient les exigences qui doivent être satisfaites pour qu'une politique de clé prenne effet. Avec des conditions, AWS peut évaluer le contexte d'une demande d'API afin de déterminer si la déclaration de politique s'applique ou non.

Pour définir les conditions, vous utilisez des clés de condition prédéfinies. AWS KMS prend en charge les clés de condition AWS globales et les clés de AWS KMS condition. Pour prendre en charge le contrôle d'accès basé sur les attributs (ABAC), AWS KMS fournit des clés de condition qui contrôlent l'accès à une clé KMS en fonction de balises et d'alias. Pour en savoir plus, consultez ABAC pour AWS KMS.

Le format d'une condition est le suivant :

```
"Condition": {"condition operator": {"condition key": "condition value"}}
```

comme :

```
"Condition": {"StringEquals": {"kms:CallerAccount": "111122223333"}}
```

Pour plus d'informations sur la syntaxe des AWS politiques, consultez la référence des politiques AWS IAM dans le guide de l'utilisateur IAM.

Exemple de politique de clé

L'exemple suivant illustre une politique de clé complète pour une clé KMS de chiffrement symétrique. Vous pouvez l'utiliser à titre de référence lorsque vous lisez les principaux concepts de stratégie de ce chapitre. Cette politique de clé combine les exemples d'instruction de politique issus de la section précédente politique de clé par défaut en une politique de clé unique qui réalise les opérations suivantes :

- Permet à l'exemple Compte AWS 111122223333 un accès complet à la clé KMS. Cela permet au compte et à ses administrateurs, y compris l'utilisateur root du compte (pour les urgences), d'utiliser des politiques IAM dans le compte pour autoriser l'accès à la clé KMS.
- Permet au rôle IAM `ExampleAdminRole` d'administrer la clé KMS.
- Permet au rôle IAM `ExampleUserRole` d'utiliser la clé KMS.

```
{
    "Id": "key-consolepolicy",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Enable IAM user Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Sid": "Allow access for Key Administrators",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/ExampleAdminRole"
            },
            "Action": [
                "kms:Create*",
                "kms:Describe*",
                "kms:Enable*",
                "kms:List*",
                "kms:Put*",
                "kms:Update*",
```

```
                "kms:Revoke*",
                "kms:Disable*",
                "kms:Get*",
                "kms:Delete*",
                "kms:TagResource",
                "kms:UntagResource",
                "kms:ScheduleKeyDeletion",
                "kms:CancelKeyDeletion",
                "kms:RotateKeyOnDemand"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/ExampleUserRole"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Allow attachment of persistent resources",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/ExampleUserRole"
            },
            "Action": [
                "kms:CreateGrant",
                "kms:ListGrants",
                "kms:RevokeGrant"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
```

```
        }
    ]
}
```

politique de clé par défaut

Lorsque vous créez une clé KMS, vous pouvez spécifier la politique de clé pour la nouvelle clé KMS. Si vous n'en fournissez pas, AWS KMS créez-en un pour vous. La politique de clé par défaut AWS KMS utilisée varie selon que vous créez la clé dans la AWS KMS console ou que vous utilisez l' AWS KMS API.

politique de clé par défaut lorsque vous créez une clé KMS par programmation

Lorsque vous créez une clé KMS par programmation avec l'AWS KMS API (y compris en utilisant le AWS Command Line Interfaceou Outils AWS pour PowerShell) AWS SDKs, et que vous ne spécifiez aucune politique de clé, AWS KMS applique une politique de clé par défaut très simple. Cette politique de clé par défaut comporte une déclaration de politique qui autorise le Compte AWS propriétaire de la clé KMS à utiliser les politiques IAM pour autoriser l'accès à toutes les AWS KMS opérations sur la clé KMS. Pour plus d'informations sur cette instruction de politique, consultez la rubrique Autorise l'accès au Compte AWS et active les politiques IAM..

Politique de clé par défaut lorsque vous créez une clé KMS avec AWS Management Console

Lorsque vous créez une clé KMS avec le AWS Management Console, la politique clé commence par la déclaration de politique qui autorise l'accès aux politiques IAM Compte AWS et les active. La console ajoute ensuite une instruction d'administrateur clé, une déclaration d'utilisateur clé et (pour la plupart des types de clés) une instruction qui permet aux principaux d'utiliser la clé KMS avec d'autres AWS services. Vous pouvez utiliser les fonctionnalités de la AWS KMS console pour spécifier les utilisateurs IAM IAMroles, et Comptes AWS qui sont les principaux administrateurs et ceux qui le sont (ou les deux).

Autorisations

- Autorise l'accès au Compte AWS et active les politiques IAM.
- Autorise les administrateurs de clés à administrer la clé KMS
- Autorise les utilisateurs de clés à utiliser la clé KMS.
  - Permet aux utilisateurs de clés d'utiliser une clé KMS pour les opérations de chiffrement
  - Permet aux utilisateurs de clé d'utiliser la clé KMS avec les services AWS .

Autorise l'accès au Compte AWS et active les politiques IAM.

L'instruction de politique de clé par défaut suivante est extrêmement importante.

- Il donne au propriétaire Compte AWS de la clé KMS un accès complet à la clé KMS.

  Contrairement aux autres politiques relatives aux AWS ressources, une politique AWS KMS clé n'autorise pas automatiquement le compte ni aucune de ses identités. Pour accorder l'autorisation aux administrateurs de compte, la politique de clé doit inclure une instruction explicite qui fournit cette autorisation, comme celle-ci.
- Celle permet au compte d'utiliser des politiques IAM pour autoriser l'accès à la clé KMS, en plus de la politique de clé.

  Sans cette autorisation, les politiques IAM qui autorisent l'accès à la clé sont inefficaces, bien que celles qui refusent l'accès à la clé soient toujours efficaces.
- Cela réduit le risque que la clé devienne ingérable en accordant une autorisation de contrôle d'accès aux administrateurs du compte, y compris l'utilisateur racine du compte, qui ne peut pas être supprimé.

L'instruction de politique de clé suivante est la politique de clé par défaut complète pour les clés KMS créées par programmation. Il s'agit de la première déclaration de politique de la politique de clé par défaut pour les clés KMS créées dans la AWS KMS console.

```
{
  "Sid": "Enable IAM User Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:root"
   },
  "Action": "kms:*",
  "Resource": "*"
}
```

Autorise les politiques IAM à autoriser l'accès à la clé KMS.

> La déclaration de politique clé présentée ci-dessus donne au Compte AWS détenteur de la clé l'autorisation d'utiliser les politiques IAM, ainsi que les politiques clés, pour autoriser toutes les actions (`kms:*`) sur la clé KMS.

Le principal dans cette instruction de politique de clé est le principal du compte, qui est représenté par un ARN au format suivant : `arn:aws:iam::account-id:root`. Le principal du compte représente le AWS compte et ses administrateurs.

Lorsque le principal d'une instruction de politique de clé correspond au principal du compte, cette instruction n'autorise aucun principal IAM à utiliser la clé KMS. Au lieu de cela, elle permet au compte d'utiliser des politiques IAM pour déléguer les autorisations spécifiées dans l'instruction de politique. Cette instruction de politique de clé par défaut permet au compte d'utiliser des politiques IAM pour déléguer l'autorisation pour toutes les actions (`kms:*`) sur la clé KMS.

réduit le risque que la clé KMS devienne ingérable.

Contrairement aux autres politiques relatives aux AWS ressources, une politique AWS KMS clé n'autorise pas automatiquement le compte ou l'un de ses principaux responsables. Pour accorder l'autorisation à un principal, y compris le principal du compte, vous devez utiliser une instruction de politique de clé qui fournit explicitement l'autorisation. Vous n'êtes pas l'obligation de donner accès à la clé KMS au principal du compte ou à tout autre principal. Cependant, donner accès au principal du compte vous permet d'éviter que la clé ne devienne ingérable.

Par exemple, supposons que vous créez une politique de clé qui donne à un seul utilisateur l'accès à la clé KMS. Si vous supprimez ensuite cet utilisateur, la clé devient ingérable et vous devez contacter le support AWS pour retrouver l'accès à la clé KMS.

La déclaration de politique clé présentée ci-dessus autorise le contrôle de la clé du compte principal, qui représente lui-même Compte AWS et ses administrateurs, y compris l'utilisateur root du compte. L'utilisateur racine du compte est le seul principal qui ne peut pas être supprimé, sauf si vous supprimez le Compte AWS. Conformément aux bonnes pratiques IAM, il est déconseillé d'agir au nom de l'utilisateur racine du compte, sauf en cas d'urgence. Toutefois, vous devrez peut-être agir en tant qu'utilisateur racine du compte si vous supprimez tous les autres utilisateurs et rôles ayant accès à la clé KMS.

Autorise les administrateurs de clés à administrer la clé KMS

La politique de clé par défaut créée par la console vous permet de choisir des utilisateurs et des rôles IAM dans le compte et d'en faire des administrateurs de clé. Cette instruction est appelée l'instruction des administrateurs de clé. Les administrateurs de clés ont les autorisations nécessaires pour gérer la clé KMS, mais n'ont pas d'autorisations pour utiliser la clé KMS dans des opérations de chiffrement. Vous pouvez ajouter des utilisateurs et des rôles IAM à la liste des administrateurs de clés lorsque vous créez la clé KMS dans la vue par défaut ou la vue de la politique.

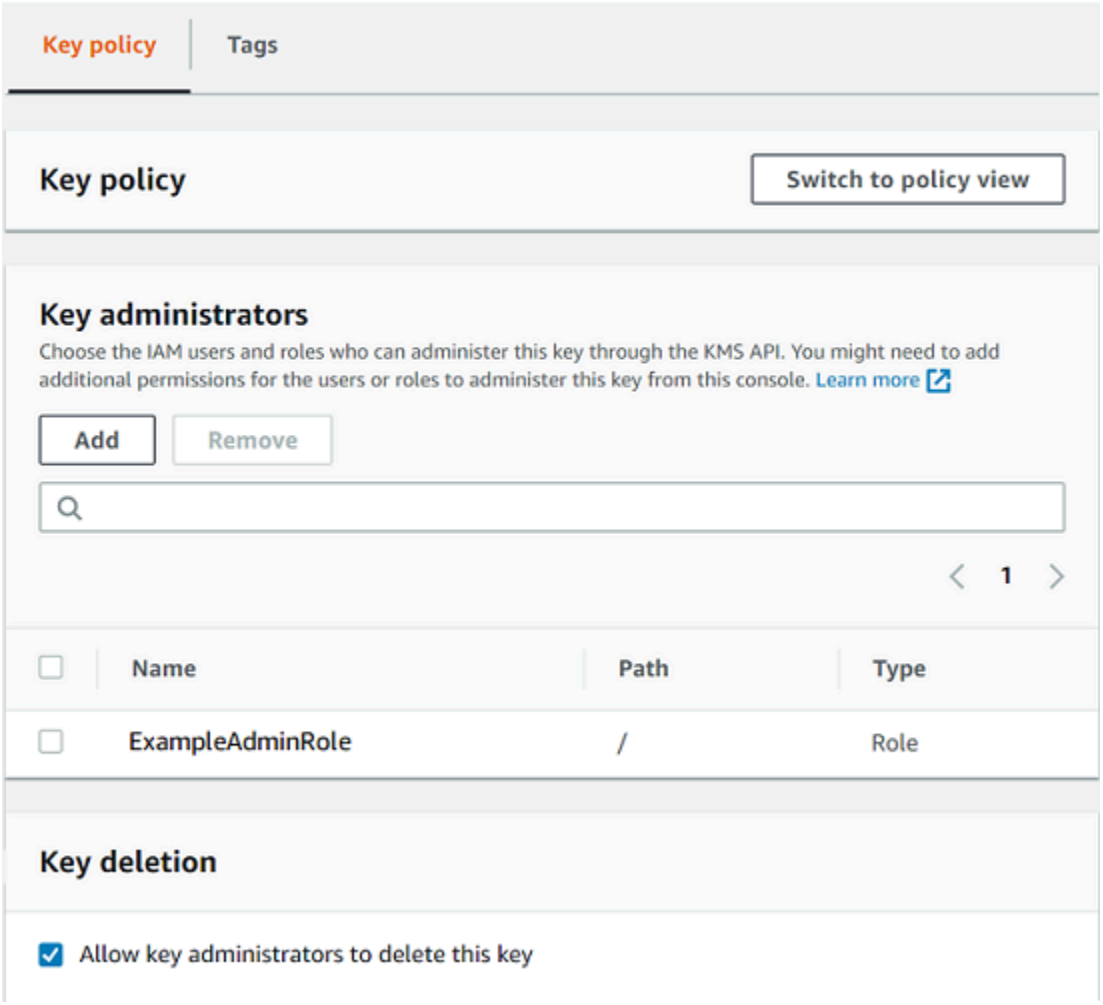
Warning

Les administrateurs clés étant autorisés à modifier la politique clé et à créer des autorisations, ils peuvent s'octroyer, ainsi qu'à d'autres, AWS KMS des autorisations non spécifiées dans cette politique.
Les principaux qui ont l'autorisation de gérer les balises et les alias peuvent également contrôler l'accès à une clé KMS. Pour en savoir plus, consultez ABAC pour AWS KMS.

Note

Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

L'exemple suivant montre l'instruction des administrateurs de clé dans la vue par défaut de la console AWS KMS .



Voici un exemple d'instruction des administrateurs de clés dans la vue de politique de la console AWS KMS . Cette déclaration des administrateurs de clés concerne une clé KMS de chiffrement symétrique à région unique.

> **Note**
>
> La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
```

```
  "Principal": {"AWS":"arn:aws:iam::111122223333:role/ExampleAdminRole"},
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:TagResource",
    "kms:UntagResource",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion",
    "kms:RotateKeyOnDemand"
  ],
  "Resource": "*"
}
```

L'instruction par défaut des administrateurs de clés pour la clé KMS la plus courante, une clé KMS de chiffrement symétrique à région unique, octroie les autorisations suivantes. Pour plus d'informations sur ces autorisations, veuillez consulter la AWS KMS autorisations.

Lorsque vous utilisez la AWS KMS console pour créer une clé KMS, la console ajoute les utilisateurs et les rôles que vous spécifiez à l'`Principal`élément figurant dans la déclaration des administrateurs clés.

Un grand nombre de ces autorisations contiennent le caractère générique (*), qui octroie toutes les autorisations commençant par le verbe spécifié. Par conséquent, lors de l' AWS KMS ajout de nouvelles opérations d'API, les administrateurs clés sont automatiquement autorisés à les utiliser. Il n'est pas nécessaire de mettre à jour vos politiques de clé pour inclure les nouvelles opérations. Si vous préférez limiter vos administrateurs de clés à un ensemble fixe d'opérations d'API, vous pouvez modifier votre politique de clé.

`kms:Create*`

Autorise `kms:CreateAlias` et `kms:CreateGrant`. (L'autorisation `kms:CreateKey` n'est valide que dans une politique IAM.)

`kms:Describe*`

Autorise `kms:DescribeKey` L'autorisation `kms:DescribeKey` est nécessaire pour afficher la page des détails d'une clé KMS dans la AWS Management Console.

`kms:Enable*`

Autorise `kms:EnableKey` Pour les clés KMS de chiffrement symétriques, elle autorise également `kms:EnableKeyRotation`.

`kms:List*`

Autorise `kms:ListGrants`, `kms:ListKeyPolicies` et `kms:ListResourceTags`. (Les autorisations `kms:ListAliases` et `kms:ListKeys`, requises pour afficher les clés KMS dans la AWS Management Console, sont valides uniquement dans les politiques IAM.)

`kms:Put*`

Autorise `kms:PutKeyPolicy` Cette autorisation autorise les administrateurs de clés à modifier la politique de cette clé KMS.

`kms:Update*`

Autorise `kms:UpdateAlias` et `kms:UpdateKeyDescription`. Pour les clés multi-région, elle autorise `kms:UpdatePrimaryRegion` sur cette clé KMS.

`kms:Revoke*`

Autorise `kms:RevokeGrant` qui permet aux administrateurs de clés de supprimer un octroi, même s'ils ne sont pas un principal de retrait dans l'octroi.

`kms:Disable*`

Autorise `kms:DisableKey` Pour les clés KMS de chiffrement symétriques, elle autorise également `kms:DisableKeyRotation`.

`kms:Get*`

Autorise `kms:GetKeyPolicy` et `kms:GetKeyRotationStatus`. Pour les clés KMS avec des éléments de clé importés, elle autorise `kms:GetParametersForImport`. Pour les clés KMS asymétriques, elle autorise `kms:GetPublicKey`. L'autorisation `kms:GetKeyPolicy` est nécessaire pour afficher la politique de clé KMS dans la AWS Management Console.

`kms:Delete*`

Autorise `kms:DeleteAlias` Pour les clés avec des éléments de clé importés, elle autorise `kms:DeleteImportedKeyMaterial`. Notez que l'autorisation `kms:Delete*` ne permet pas aux administrateurs de clés de supprimer la clé KMS (`ScheduleKeyDeletion`).

`kms:TagResource`

Autorise `kms:TagResource`, qui permet aux administrateurs de clés d'ajouter des identifications à la clé KMS. Étant donné que les balises peuvent être également utilisées pour contrôler l'accès à la clé KMS, cette autorisation peut permettre aux administrateurs d'autoriser ou de refuser l'accès à la clé KMS. Pour en savoir plus, consultez ABAC pour AWS KMS.

`kms:UntagResource`

Autorise `kms:UntagResource`, qui permet aux administrateurs de clés de supprimer les balises de la clé KMS. Étant donné que les balises peuvent être utilisées pour contrôler l'accès à la clé, cette autorisation peut permettre aux administrateurs d'autoriser ou de refuser l'accès à la clé KMS. Pour en savoir plus, consultez ABAC pour AWS KMS.

`kms:ScheduleKeyDeletion`

Autorise `kms:ScheduleKeyDeletion`, qui permet aux administrateurs de clés de supprimer cette clé KMS. Pour supprimer cette autorisation, désélectionnez l'option Autoriser les administrateurs de clé à supprimer cette clé.

`kms:CancelKeyDeletion`

Autorise `kms:CancelKeyDeletion`, qui permet aux administrateurs de clés d'annuler la suppression de cette clé KMS. Pour supprimer cette autorisation, désélectionnez l'option Autoriser les administrateurs de clé à supprimer cette clé.

`kms:RotateKeyOnDemand`

`kms:RotateKeyOnDemand`Autorise, ce qui permet aux administrateurs clés d'effectuer une rotation à la demande du contenu clé de cette clé KMS.

AWS KMS ajoute les autorisations suivantes à la déclaration par défaut des administrateurs de clés lorsque vous créez des clés spécifiques.

`kms:ImportKeyMaterial`

L'autorisation `kms:ImportKeyMaterial` permet aux administrateurs de clés d'importer des éléments de clé dans la clé KMS. Cette autorisation est incluse dans la politique de clé uniquement lorsque vous créez une clé KMS sans élément de clé.

`kms:ReplicateKey`

L'`kms:ReplicateKey`autorisation permet aux administrateurs clés de créer une réplique d'une clé primaire multirégionale dans une AWS région différente. Cette autorisation est incluse dans la politique de clé uniquement lorsque vous créez une clé primaire ou un réplica multi-région.

`kms:UpdatePrimaryRegion`

L'autorisation `kms:UpdatePrimaryRegion` permet aux administrateurs de clés de remplacer une clé de réplica multi-région par une clé primaire multi-région. Cette autorisation est incluse dans la politique de clé uniquement lorsque vous créez une clé primaire ou un réplica multi-région.

Autorise les utilisateurs de clés à utiliser la clé KMS.

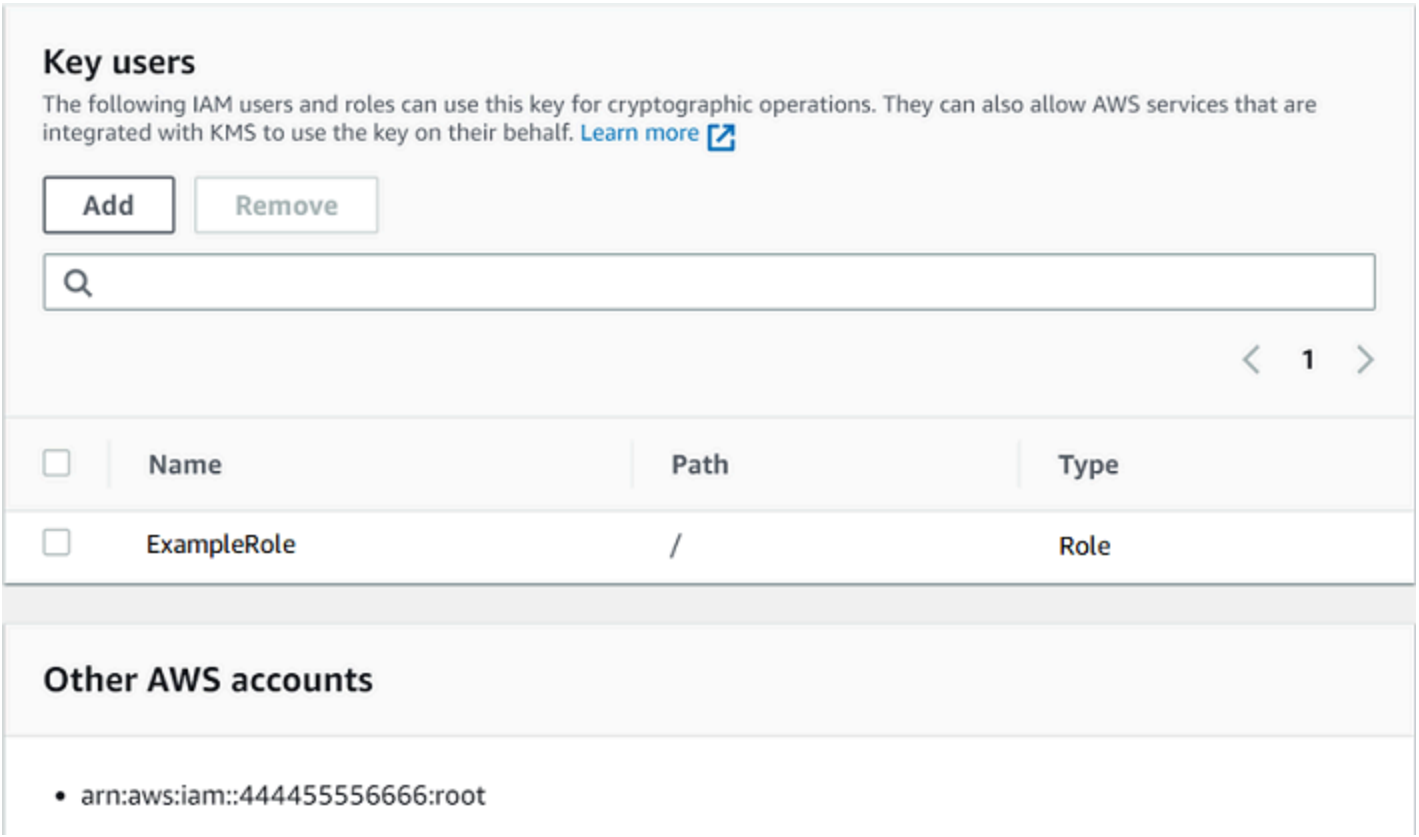
La politique de clés par défaut créée par la console pour les clés KMS vous permet de choisir les utilisateurs IAM et les rôles IAM dans le compte, ainsi que des utilisateurs externes Comptes AWS, et d'en faire des utilisateurs clés.

La console ajoute deux instructions de politique à la politique de clé pour les utilisateurs de clés.

- Utilisez la clé KMS directement – La première instruction de politique de clé donne aux utilisateurs des clés l'autorisation d'utiliser la clé KMS directement pour toutes les opérations de chiffrement prises en charge pour ce type de clé KMS.
- Utiliser la clé KMS avec les AWS services — La deuxième déclaration de politique autorise les utilisateurs clés à autoriser les AWS services intégrés AWS KMS à utiliser la clé KMS en leur nom pour protéger des ressources, telles que les compartiments Amazon S3 et les tables Amazon DynamoDB.

Vous pouvez ajouter des utilisateurs IAM, des rôles IAM et d'autres utilisateurs Comptes AWS à la liste des utilisateurs clés lorsque vous créez la clé KMS. Vous pouvez aussi modifier la liste avec la vue par défaut de la console pour les politiques de clé, comme illustré dans l'image suivante. La vue par défaut pour les politiques de clé est sur la page des détails de la clé. Pour plus d'informations sur la manière d'autoriser les utilisateurs d'autres pays Comptes AWS à utiliser la clé KMS, consultezAutoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

> **Note**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.



Les instructions par défaut des utilisateurs de clé d'une clé symétrique à région unique octroient les autorisations suivantes. Pour plus d'informations sur ces autorisations, veuillez consulter la AWS KMS autorisations.

Lorsque vous utilisez la AWS KMS console pour créer une clé KMS, la console ajoute les utilisateurs et les rôles que vous spécifiez à l'`Principal`élément figurant dans la déclaration de chaque utilisateur clé.

> **Note**
>
> La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:role/ExampleRole",
    "arn:aws:iam::444455556666:root"
  ]},
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*"
},
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:role/ExampleRole",
    "arn:aws:iam::444455556666:root"
  ]},
  "Action": [
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

Permet aux utilisateurs de clés d'utiliser une clé KMS pour les opérations de chiffrement

Les utilisateurs de clés ont l'autorisation d'utiliser la clé KMS directement dans toutes les opérations de chiffrement prises en charge par la clé KMS. Ils peuvent également utiliser l'DescribeKeyopération pour obtenir des informations détaillées sur la clé KMS dans la AWS KMS console ou en utilisant les opérations AWS KMS d'API.

Par défaut, la AWS KMS console ajoute à la politique clé par défaut des instructions relatives aux utilisateurs clés telles que celles des exemples suivants. Étant donné qu'ils prennent en charge différentes opérations d'API, les actions des instructions de politique pour les clés KMS de chiffrement symétriques, les clés KMS HMAC, les clés KMS asymétriques pour le chiffrement de clé publique et les clés KMS asymétriques pour la signature et la vérification sont légèrement différentes.

Clés KMS de chiffrement symétriques

La console ajoute l'instruction suivante à la politique de clé pour les clés KMS de chiffrement symétriques.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey",
    "kms:Encrypt",
    "kms:GenerateDataKey*",
    "kms:ReEncrypt*"
  ],
  "Resource": "*"
}
```

Clés KMS HMAC

La console ajoute l'instruction suivante à la politique de clé pour les clés KMS HMAC.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
```

```
    "kms:DescribeKey",
    "kms:GenerateMac",
    "kms:VerifyMac"
  ],
  "Resource": "*"
}
```

Clés CMK asymétriques pour le chiffrement de clé publique

La console ajoute l'instruction suivante à la politique de clé pour les clés KMS asymétriques avec Encrypt and decrypt (Chiffrer et déchiffrer) comme utilisation des clés.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:DescribeKey",
    "kms:GetPublicKey"
  ],
  "Resource": "*"
}
```

Clés CMK asymétriques pour la signature et la vérification

La console ajoute l'instruction suivante à la politique de clé pour les clés KMS asymétriques avec Sign and verify (Signer et vérifier) comme utilisation des clés.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:DescribeKey",
    "kms:GetPublicKey",
    "kms:Sign",
    "kms:Verify"
  ],
```

```
  "Resource": "*"
}
```

Clés KMS asymétriques pour dériver des secrets partagés

La console ajoute la déclaration suivante à la politique clé pour les clés KMS asymétriques avec une utilisation clé de l'accord des clés.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:DescribeKey",
    "kms:GetPublicKey",
    "kms:DeriveSharedSecret"
  ],
  "Resource": "*"
}
```

Les actions de ces instructions donnent aux utilisateurs de clé les autorisations suivantes.

`kms:Encrypt`

Permet aux utilisateurs de clé de chiffrer des données avec cette clé KMS.

`kms:Decrypt`

Permet aux utilisateurs de clé de déchiffrer des données avec cette clé KMS.

`kms:DeriveSharedSecret`

Permet aux utilisateurs clés de dériver des secrets partagés avec cette clé KMS.

`kms:DescribeKey`

Permet aux utilisateurs de clé d'obtenir des informations sur cette clé KMS, y compris ses identificateurs, sa date de création, son état, etc. Il permet également aux principaux utilisateurs d'afficher les détails de la clé KMS dans la AWS KMS console.

`kms:GenerateDataKey*`

Permet aux utilisateurs de clé de demander une paires de clés de données symétriques ou asymétriques pour les opérations de chiffrement côté client. La console utilise

le caractère générique * pour représenter l'autorisation pour les opérations d'API suivantes : GenerateDataKey, GenerateDataKeyWithoutPlaintextGenerateDataKeyPair, et GenerateDataKeyPairWithoutPlaintext. Ces autorisations sont valides uniquement sur les clés KMS symétriques qui chiffrent les clés de données.

km : GenerateMac

Permet aux utilisateurs de clés d'utiliser une clé KMS HMAC pour générer une balise HMAC.

km : GetPublicKey

Permet aux utilisateurs de clé de télécharger la clé publique de la clé KMS asymétrique. Les parties avec lesquelles vous partagez cette clé publique peuvent chiffrer des données en dehors de AWS KMS. Cependant, ces textes chiffrés ne peuvent être déchiffrés qu'en appelant l'opération Decrypt dans AWS KMS.

km : ReEncrypt *

Permet aux utilisateurs de clé de rechiffrer les données initialement chiffrées avec cette clé KMS, ou d'utiliser cette clé KMS pour rechiffrer des données précédemment chiffrées. L'ReEncryptopération nécessite l'accès aux clés KMS source et de destination. Pour ce faire, vous pouvez accorder l'autorisation `kms:ReEncryptFrom` sur la clé KMS source et l'autorisation `kms:ReEncryptTo` sur la clé KMS de destination. Cependant, par souci de simplicité, la console autorise `kms:ReEncrypt*` (avec le caractère * générique) sur les deux clés KMS.

kms:Sign

Permet aux utilisateurs de clé de signer des messages avec cette clé KMS.

kms:Verify

Permet aux utilisateurs de clé de vérifier les signatures avec cette clé KMS.

km : VerifyMac

Permet aux utilisateurs de clés d'utiliser une clé KMS HMAC pour vérifier une balise HMAC.

Permet aux utilisateurs de clé d'utiliser la clé KMS avec les services AWS .

La politique clé par défaut de la console donne également aux utilisateurs clés les autorisations dont ils ont besoin pour protéger leurs données dans les AWS services utilisant des autorisations. AWS les services utilisent souvent des subventions pour obtenir une autorisation spécifique et limitée d'utilisation d'une clé KMS.

Cette déclaration de politique clé permet à l'utilisateur clé de créer, de consulter et de révoquer des autorisations sur la clé KMS, mais uniquement lorsque la demande d'opération d'autorisation provient d'un AWS service intégré à AWS KMS. La condition de GrantIsFor AWSResource politique kms : ne permet pas à l'utilisateur d'appeler directement ces opérations de subvention. Lorsque l'utilisateur clé l'autorise, un AWS service peut créer une autorisation au nom de l'utilisateur qui permet au service d'utiliser la clé KMS pour protéger les données de l'utilisateur.

Les utilisateurs de clé ont besoin de ces autorisations pour utiliser leur clé KMS avec des services intégrés, mais ces autorisations ne sont pas suffisantes. Les utilisateurs de clés ont également besoin d'une autorisation pour utiliser les services intégrés. Pour en savoir plus sur l'accès des utilisateurs à un AWS service intégré AWS KMS, consultez la documentation du service intégré.

```
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

Par exemple, les utilisateurs de clés peuvent utiliser ces autorisations sur la clé KMS de la manière suivante.

- Utilisez cette clé KMS avec Amazon Elastic Block Store (Amazon EBS) et Amazon Elastic Compute Cloud (EC2Amazon) pour associer un volume EBS chiffré à une instance. EC2 L'utilisateur clé EC2 autorise implicitement Amazon à utiliser la clé KMS pour associer le volume chiffré à l'instance. Pour de plus amples informations, veuillez consulter Comment Amazon Elastic Block Store (Amazon EBS) utilise AWS KMS.
- Utilisez cette clé KMS avec Amazon Redshift pour lancer un cluster chiffré. L'utilisateur de clé accorde implicitement à Amazon Redshift l'autorisation d'utiliser la clé KMS pour lancer le cluster chiffré et créer des instantanés chiffrés. Pour de plus amples informations, veuillez consulter Comment Amazon Redshift utilise AWS KMS.
- Utiliser cette clé KMS avec d'autres services AWS intégrés à AWS KMS qui utilisent des octrois, pour créer, gérer ou utiliser des ressources chiffrées avec ces services.

La politique de clé par défaut permet aux utilisateurs clés de déléguer leur autorisation d'octroi à tous les services intégrés qui utilisent des octrois. Cependant, vous pouvez créer une politique clé personnalisée qui limite l'autorisation à des AWS services spécifiques. Pour plus d'informations, reportez-vous à la clé de condition km : ViaService.

Afficher une politique clé

Vous pouvez consulter la politique clé d'une clé gérée par le AWS KMS client ou d'une clé Clé gérée par AWSintégrée à votre compte à l'aide de la AWS KMS console ou de l'GetKeyPolicyopération dans l' AWS KMS API. Vous ne pouvez pas utiliser ces techniques pour afficher la politique d'une clé KMS dans un autre Compte AWS.

Pour en savoir plus sur les AWS KMS principales politiques, voirPolitiques clés en AWS KMS. Pour savoir comment déterminer quels sont les utilisateurs et rôles qui ont accès à une clé KMS, veuillez consulter the section called “Détermination de l'accès”.

Utilisation de la AWS KMS console

Les utilisateurs autorisés peuvent afficher la politique de clé d'une Clé gérée par AWS ou d'une clé gérée par le client dans l'onglet Politique de clé de la AWS Management Console.

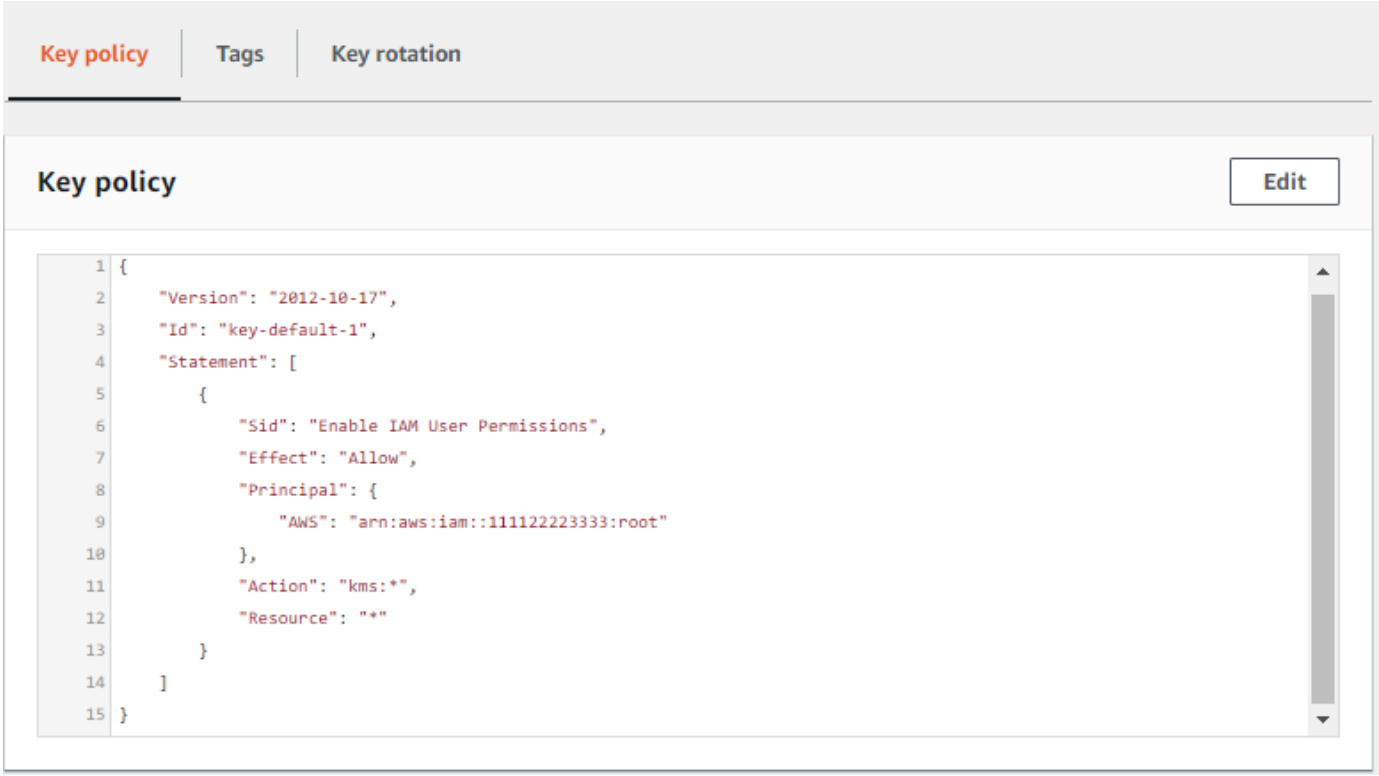
Pour afficher la politique clé d'une clé KMS dans le AWS Management Console, vous devez disposer des GetKeyPolicy autorisations kms : DescribeKey, kms : et kms :. ListAliases

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Pour afficher les clés de votre compte qui AWS crée et gère pour vous, dans le volet de navigation, choisissez les clés AWS gérées. Pour afficher les clés de votre compte que vous créez et gérez vous-même, dans le volet de navigation, choisissez Clés gérées par le client.
4. Dans la liste des clés KMS, choisissez l'alias ou l'ID de clé de la clé KMS que vous souhaitez examiner.
5. Choisissez l'onglet Politique de clé.

   Dans l'onglet Politique de clé, vous pouvez voir le document de politique de clé. Il s'agit d'une vue de politique. Dans les instructions de politique de clé, vous pouvez voir les principaux qui

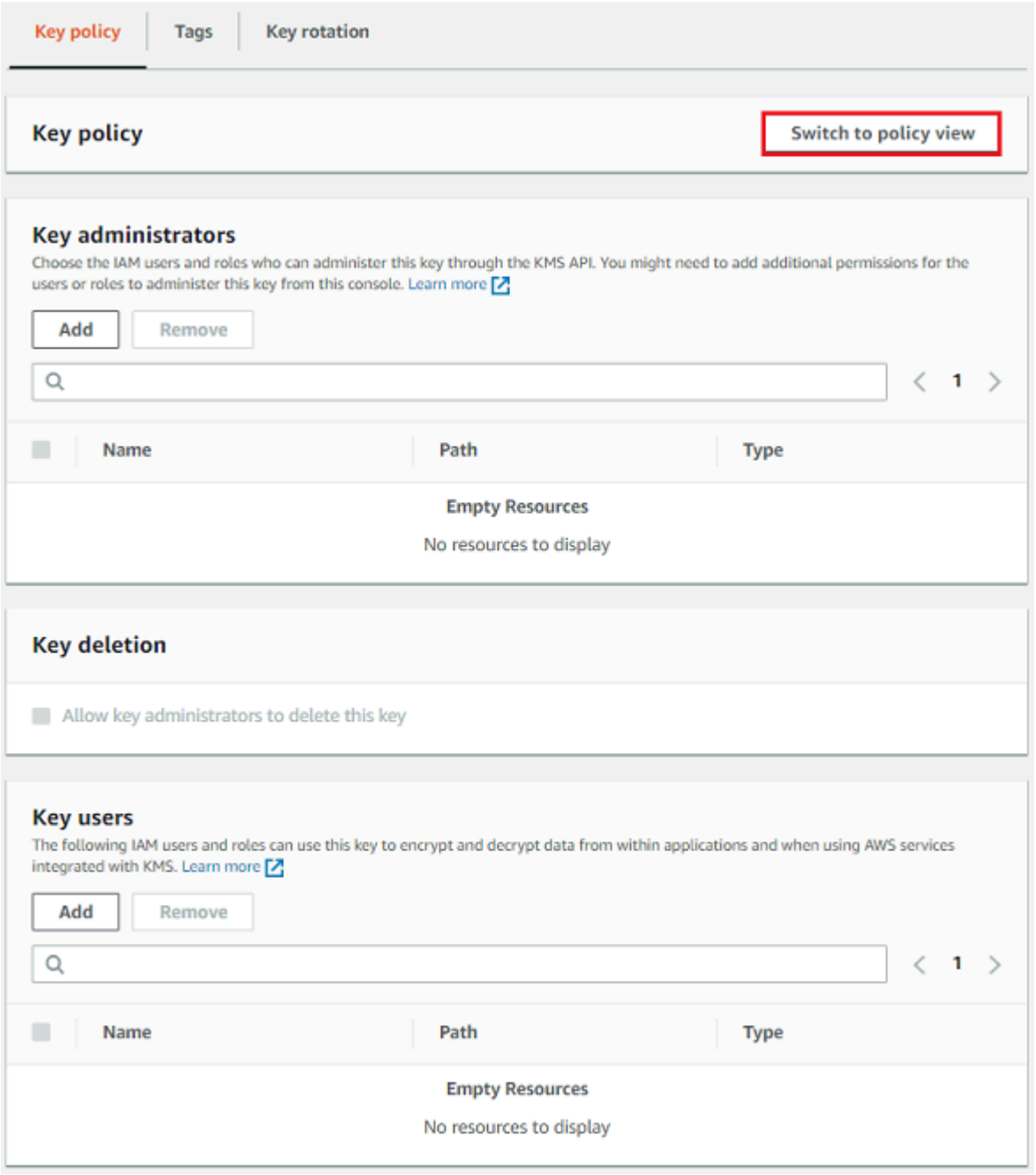
sont autorisés à accéder à la clé KMS par la politique de clé, ainsi que les actions qu'ils peuvent effectuer.

L'exemple suivant montre la vue de la politique de clé par défaut.



Ou, si vous avez créé la clé KMS dans le AWS Management Console, vous verrez la vue par défaut avec des sections pour les administrateurs clés, la suppression de clés et les utilisateurs clés. Pour consulter le document de politique de clé, choisissez Passer à la vue de politique.

L'exemple suivant montre la vue par défaut de la politique de clé par défaut.



Utilisation de l' AWS KMS API

Pour obtenir la politique de clé pour une clé KMS dans votre Compte AWS, utilisez l'GetKeyPolicyopération dans l' AWS KMS API. Vous ne pouvez pas utiliser cette opération pour afficher une politique de clé d'un autre compte.

L'exemple suivant utilise la get-key-policycommande contenue dans le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel AWS SDK pour effectuer cette demande.

Notez que le paramètre `PolicyName` est obligatoire, même si `default` est sa seule valeur valide. En outre, cette commande demande une sortie en texte, plutôt qu'en JSON, pour le rendre plus facile à afficher.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un identifiant valide provenant de votre compte.

```
$ aws kms get-key-policy --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --policy-name
 default --output text
```

La réponse doit être similaire à la suivante, qui renvoie la politique de clé par défaut.

```
{
  "Version" : "2012-10-17",
  "Id" : "key-consolepolicy-3",
  "Statement" : [ {
    "Sid" : "Enable IAM User Permissions",
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "arn:aws:iam::111122223333:root"
    },
    "Action" : "kms:*",
    "Resource" : "*"
  } ]
}
```

Modifier une politique clé

Vous pouvez modifier la politique de clé d'une clé KMS dans votre ordinateur en Compte AWS utilisant l'PutKeyPolicyopération AWS Management Console ou. Vous ne pouvez pas utiliser ces techniques pour modifier la politique clé d'une clé KMS d'un autre Compte AWS.

Lors de la modification d'une politique de clé, gardez à l'esprit les règles suivantes :

- Vous pouvez afficher la politique de clé d'une Clé gérée par AWS ou d'une clé KMS par le client, mais vous ne pouvez modifier que la politique de clé d'une clé KMS géré par le client. Les politiques de Clés gérées par AWS sont créées et gérées par le AWS service qui a créé la clé KMS dans votre compte. Vous ne pouvez pas afficher ou modifier la politique de clé pour une Clé détenue par AWS.

- Vous pouvez ajouter ou supprimer des utilisateurs IAM, des rôles IAM et Comptes AWS dans la politique clé, et modifier les actions autorisées ou refusées pour ces principaux. Pour plus d'informations sur les moyens de spécifier des principaux et des autorisations dans une politique de clé, consultez la page Politiques de clé.
- Vous ne pouvez pas ajouter de groupes IAM à une politique de clé, mais vous pouvez ajouter plusieurs utilisateurs IAM et rôles IAM. Pour de plus amples informations, veuillez consulter Attribution à plusieurs principaux IAM de l'autorisation d'accès à une clé KMS.
- Si vous ajoutez des éléments externes Comptes AWS à une politique clé, vous devez également utiliser des politiques IAM dans les comptes externes pour accorder des autorisations aux utilisateurs, aux groupes ou aux rôles IAM dans ces comptes. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.
- Le document de politique de clé obtenu ne doit pas comporter plus de 32 Ko (32 768 octets).

Comment modifier une politique de clé

Vous pouvez modifier une politique de clé de trois façons différentes, chacune d'elles étant expliquée dans les sections suivantes.

Rubriques

- Utilisation de la vue par défaut d' AWS Management Console
- Utilisation de l'affichage AWS Management Console des politiques
- Utilisation de l' AWS KMS API

Utilisation de la vue par défaut d' AWS Management Console

Vous pouvez utiliser la console pour modifier une politique de clé à l'aide d'une interface graphique appelée vue par défaut.

Si les étapes suivantes ne correspondent pas à ce que vous voyez dans la console, cela peut signifier que cette politique n'a pas été créée par la console. Ou cela peut signifier que la politique de clé a été modifiée d'une façon que la vue par défaut de la console ne prend pas en charge. Dans ce cas, suivez les étapes de la section Utilisation de l'affichage AWS Management Console des politiques ou Utilisation de l' AWS KMS API.

1. Affichez la politique de clé d'une clé gérée par le client comme indiqué dans Utilisation de la AWS KMS console. (Vous ne pouvez pas modifier les politiques clés de Clés gérées par AWS.)

2. Décidez ce qu'il convient de modifier.

   - Pour ajouter ou supprimer des administrateurs de clé, et pour autoriser ou non ces administrateurs de clé à supprimer la clé KMS, utilisez les contrôles de la section Administrateurs de clé de la page. Les administrateurs de clé gèrent la clé KMS, y compris son activation et sa désactivation, la définition de la politique de clé, et l'activation de la rotation des clés.
   - Pour ajouter ou supprimer des utilisateurs clés, et pour autoriser ou interdire Comptes AWS à des utilisateurs externes d'utiliser la clé KMS, utilisez les commandes de la section Utilisateurs clés de la page. Les utilisateurs de clé peuvent utiliser la clé KMS dans les opérations de chiffrement, telles que le chiffrement, le déchiffrement, le rechiffrement et la génération de clés de données.

Utilisation de l'affichage AWS Management Console des politiques

Vous pouvez utiliser la console pour modifier un document de politique de clé à l'aide de la vue de politique de la console.

1. Affichez la politique de clé d'une clé gérée par le client comme indiqué dans Utilisation de la AWS KMS console. (Vous ne pouvez pas modifier les politiques clés de Clés gérées par AWS.)
2. Dans la section Politique de clé, choisissez Passer à la vue de la politique.
3. Choisissez Modifier.
4. Décidez ce qu'il convient de modifier.

   - Pour ajouter un nouveau relevé, choisissez Ajouter un nouveau relevé. Vous pouvez ensuite sélectionner les actions, les principes et les conditions de votre nouvelle déclaration de politique clé parmi les options répertoriées dans le panneau du générateur de déclarations, ou saisir manuellement les éléments de la déclaration de politique.
   - Pour supprimer une déclaration de votre politique clé, sélectionnez-la, puis choisissez Supprimer. Passez en revue la déclaration de politique sélectionnée et confirmez que vous souhaitez la supprimer. Si vous décidez de ne pas procéder à la suppression de l'instruction, choisissez Annuler.
   - Pour modifier une déclaration de politique clé existante, sélectionnez-la. Vous pouvez ensuite utiliser le panneau du générateur de relevés pour choisir les éléments spécifiques que vous souhaitez modifier ou modifier manuellement le relevé.
5. Sélectionnez Enregistrer les modifications.

Utilisation de l' AWS KMS API

Vous pouvez utiliser cette PutKeyPolicyopération pour modifier la politique de clé d'une clé KMS dans votre Compte AWS. Vous ne pouvez pas utiliser cette API sur une clé KMS d'un autre Compte AWS.

1. Utilisez cette GetKeyPolicyopération pour obtenir le document de stratégie clé existant, puis enregistrez le document de stratégie clé dans un fichier. Pour obtenir un exemple de code dans plusieurs langages de programmation, veuillez consulter Utilisation `GetKeyPolicy` avec un AWS SDK ou une CLI.
2. Ouvrez le document de politique de clé dans votre éditeur de texte préféré, modifiez le document de politique de clé, puis enregistrez le fichier.
3. Utilisez cette PutKeyPolicyopération pour appliquer le document de politique clé mis à jour à la clé KMS. Pour obtenir un exemple de code dans plusieurs langages de programmation, veuillez consulter Utilisation `PutKeyPolicy` avec un AWS SDK ou une CLI.

Pour un exemple de copie d'une politique clé d'une clé KMS à une autre, consultez l'GetKeyPolicy exemple dans le manuel de référence des AWS CLI commandes.

Autorisations pour les AWS services dans les politiques clés

De nombreux AWS services utilisent AWS KMS keys pour protéger les ressources qu'ils gèrent. Lorsqu'un service utilise Clés détenues par AWS ou Clés gérées par AWS, le service établit et gère les politiques de clé de ces clés KMS.

Toutefois, lorsque vous utilisez une clé gérée par le client avec un service AWS vous définissez et gérez la politique de clé. Cette politique de clé doit accorder au service les autorisations minimales dont il a besoin pour protéger la ressource en votre nom. Nous vous recommandons de respecter le principe du moindre privilège : ne donnez au service que les autorisations dont il a besoin. Vous pouvez le faire efficacement en déterminant de quelles autorisations le service a besoin et en utilisant des clés de condition globales AWS et des clés de condition AWS KMS pour affiner les autorisations.

Pour trouver les autorisations requises par le service sur une clé gérée par le client, consultez la documentation sur le chiffrement du service. Par exemple, pour connaître les autorisations requises par Amazon Elastic Block Store (Amazon EBS), consultez la section Permissions pour les utilisateurs IAM dans le guide de l'utilisateur Amazon et le guide de EC2 l'utilisateur Amazon. EC2 Pour connaître les autorisations requises par Secrets Manager, consultez Autorisation de l'utilisation de la clé KMS dans le guide de l'utilisateur AWS Secrets Manager .

Utilisation des politiques IAM avec AWS KMS

Vous pouvez utiliser des politiques IAM, ainsi que des politiques clés, des autorisations et des politiques de point de terminaison VPC, pour contrôler l'accès à AWS KMS keys votre entrée. AWS KMS

> **Note**
>
> Pour utiliser une politique IAM afin de contrôler l'accès à une clé KMS, la politique de clé de la clé KMS doit donner au compte l'autorisation d'utiliser des politiques IAM. Plus précisément, la politique de clé doit inclure l'instruction de politique qui autorise les politiques IAM.
> Cette section explique comment utiliser les politiques IAM pour contrôler l'accès aux AWS KMS opérations. Pour plus d'informations sur IAM, veuillez consulter le Guide de l'utilisateur IAM.

Toutes les clés KMS doivent avoir une politique de clé. Les politiques IAM sont facultatives. Pour utiliser une politique IAM afin de contrôler l'accès à une clé KMS, la politique de clé de la clé KMS doit donner au compte l'autorisation d'utiliser des politiques IAM. Plus précisément, la politique de clé doit inclure l'instruction de politique qui autorise les politiques IAM.

Les politiques IAM peuvent contrôler l'accès à n'importe quelle AWS KMS opération. Contrairement aux politiques clés, les politiques IAM peuvent contrôler l'accès à plusieurs clés KMS et fournir des autorisations pour les opérations de plusieurs AWS services connexes. Mais les politiques IAM sont particulièrement utiles pour contrôler l'accès aux opérations, par exemple celles CreateKeyqui ne peuvent pas être contrôlées par une politique clé car elles n'impliquent aucune clé KMS en particulier.

Si vous accédez AWS KMS via un point de terminaison Amazon Virtual Private Cloud (Amazon VPC), vous pouvez également utiliser une politique de point de terminaison VPC pour limiter l'accès à vos AWS KMS ressources lorsque vous utilisez le point de terminaison. Par exemple, lorsque vous utilisez le point de terminaison VPC, vous pouvez uniquement autoriser les principaux utilisateurs Compte AWS à accéder à vos clés gérées par le client. Pour plus de détails, consultez la section Politiques relatives aux points de terminaison VPC.

Pour obtenir de l'aide sur la rédaction et la mise en forme d'un document de politique JSON, veuillez consulter Référence de politique JSON IAM dans le Guide de l'utilisateur IAM.

Vous pouvez utiliser les politiques IAM comme suit :

- Associer une politique d'autorisations à un rôle pour les autorisations de fédération ou entre comptes : vous pouvez associer une politique IAM à un rôle IAM pour activer la fédération d'identités, autoriser les autorisations entre comptes ou accorder des autorisations aux applications exécutées sur des instances. EC2 Pour plus d'informations sur les différents cas d'utilisation pour les rôles IAM, veuillez consulter Rôles IAM dans le Guide de l'utilisateur IAM.
- Attribuez une politique d'autorisations à un utilisateur ou à un groupe - Vous pouvez attribuer à un utilisateur ou à un groupe d'utilisateurs une politique qui les autorise à appeler des opérations AWS KMS . Toutefois, chaque fois que possible, les bonnes pratiques IAM recommandent d'utiliser des identités dotées d'informations d'identification temporaires, comme des rôles IAM.

L'exemple suivant montre une politique IAM avec des AWS KMS autorisations. Cette politique autorise les identités IAM auxquelles elle est attachée à répertorier toutes les clés KMS et leurs alias.

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

Comme toutes les politiques IAM, elle n'a pas d'élément `Principal`. Lorsque vous attribuez une politique IAM à une identité IAM, cette identité bénéficie des autorisations spécifiées dans la politique.

Pour un tableau présentant toutes les actions d' AWS KMS API et les ressources auxquelles elles s'appliquent, consultez leRéférence des autorisations .

Attribution à plusieurs principaux IAM de l'autorisation d'accès à une clé KMS

Les groupes IAM ne sont pas des principaux valides dans une politique de clé. Pour autoriser plusieurs utilisateurs et rôles à accéder à une clé KMS, effectuez l'une des opérations suivantes :

- Utilisez un rôle IAM comme principal dans la politique de clé. Plusieurs utilisateurs autorisés peuvent assumer ce rôle selon les besoins. Pour plus d'informations, consultez Rôles IAM dans le Guide de l'utilisateur IAM.

  Rien ne vous empêche d'associer plusieurs utilisateurs IAM à une politique de clé, mais cette pratique est déconseillée, car elle vous oblige à mettre à jour la politique de clé chaque fois que la liste des utilisateurs autorisés change. En outre, les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
- Utilisez une politique IAM pour accorder une autorisation à un groupe IAM. Pour ce faire, assurez-vous que la politique de clé contient la déclaration qui permet aux politiques IAM d'autoriser l'accès à la clé KMS, créez une politique IAM qui autorise l'accès à la clé KMS, puis attribuez cette politique à un groupe IAM dans lequel figurent les utilisateurs IAM autorisés. Grâce à cette approche, vous n'avez pas besoin de modifier de politiques lorsque la liste des utilisateurs autorisés change. Au lieu de cela, il vous suffit d'ajouter ou de supprimer ces utilisateurs à partir du groupe IAM approprié. Pour plus d'informations, consultez Groupes d'utilisateurs IAM dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur la manière dont les politiques AWS KMS clés et les politiques IAM fonctionnent ensemble, consultezAWS KMS Permissions de résolution des.

Bonnes pratiques pour les politiques IAM

La sécurisation de l'accès AWS KMS keys est essentielle à la sécurité de toutes vos AWS ressources. Les clés KMS sont utilisées pour protéger la plupart des ressources les plus sensibles de votre Compte AWS. Prenez le temps de concevoir les politiques clés, les politiques IAM, les autorisations et les politiques de point de terminaison VPC qui contrôlent l'accès à vos clés KMS.

Dans les instructions de politique IAM qui contrôlent l'accès aux clés KMS, utilisez le principe du moindre privilège. N'accordez aux principaux IAM que les autorisations dont ils ont besoin pour les clés KMS qu'ils doivent utiliser ou gérer.

Les meilleures pratiques suivantes s'appliquent aux politiques IAM qui contrôlent l'accès aux AWS KMS clés et aux alias. Pour obtenir des conseils généraux sur les bonnes pratiques en matière de politique IAM, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Utilisation de politiques de clé

Dans la mesure du possible, fournissez des autorisations dans les politiques de clé qui affectent une clé KMS, plutôt que dans une politique IAM qui peut s'appliquer à de nombreuses clés KMS, y compris celles d'autres Comptes AWS. Cela est particulièrement important pour les autorisations sensibles telles que kms : PutKeyPolicy et kms : ScheduleKeyDeletion mais également pour les opérations cryptographiques qui déterminent la manière dont vos données sont protégées.

Limiter CreateKey l'autorisation

Donnez l'autorisation de créer des clés (kms : CreateKey) uniquement aux principaux qui en ont besoin. Les principaux qui créent une clé KMS définissent également sa politique de clé, afin qu'ils puissent se donner, ainsi qu'à d'autres, l'autorisation d'utiliser et de gérer les clés KMS qu'ils créent. Lorsque vous accordez cette autorisation, envisagez de la limiter en utilisant les conditions de politique. Par exemple, vous pouvez utiliser la KeySpec condition kms : pour limiter l'autorisation aux clés KMS de chiffrement symétriques.

Spécifier des clés KMS dans une politique IAM

La bonne pratique consiste à spécifier l'ARN de clé de chaque clé KMS à laquelle l'autorisation s'applique dans l'élément `Resource` de l'instruction de politique. Cette pratique limite l'autorisation aux clés KMS requises par le principal. Par exemple, cet élément `Resource` ne répertorie que les clés KMS que le principal doit utiliser.

```
"Resource": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
]
```

Lorsque la spécification de clés KMS n'est pas pratique, utilisez une `Resource` valeur qui limite l'accès aux clés KMS dans une région sécurisée Compte AWS , telle que`arn:aws:kms:`*region*`:`*account*`:key/*`. Ou limitez l'accès aux clés KMS dans toutes les régions (*) d'une région fiable Compte AWS, telle que`arn:aws:kms:*:`*account*`:key/*`.

Vous ne pouvez pas utiliser d'ID de clé, de nom d'alias ou d'ARN d'alias pour représenter une clé KMS dans le champ `Resource` d'une politique IAM. Si vous spécifiez un ARN d'alias, la politique s'applique à l'alias et non à la clé KMS. Pour plus d'informations sur les politiques IAM d'alias, veuillez consulter Contrôle de l'accès aux alias.

Évitez « Resource » : « * » dans une politique IAM.

Utilisez judicieusement les caractères génériques (*). Dans une politique de clé, le caractère générique de l'élément `Resource` représente la clé KMS à laquelle la politique de clé est attachée. Mais dans une politique IAM, seul un caractère générique dans l'`Resource`eélément (`"Resource": "*"`) applique les autorisations à toutes les clés KMS Comptes AWS que le compte principal est autorisé à utiliser. Cela peut inclure des clés KMS dans d'autres Comptes AWS, ainsi que des clés KMS dans le compte du principal.

Par exemple, pour utiliser une clé KMS dans un autre compte Compte AWS, un principal doit obtenir l'autorisation de la politique de clé KMS du compte externe et de la politique IAM de son propre compte. Supposons qu'un compte arbitraire ait donné à votre Compte AWS l'autorisation kms:Decrypt sur leurs clés KMS. Si c'est le cas, une politique IAM de votre compte qui donne à un rôle l'autorisation `kms:Decrypt` sur toutes les clés KMS (`"Resource": "*"`) satisferait à la partie IAM de l'exigence. Par conséquent, les principaux qui peuvent endosser ce rôle peuvent désormais déchiffrer les textes chiffrés à l'aide de la clé KMS du compte non approuvé. Les entrées relatives à leurs opérations apparaissent dans les CloudTrail journaux des deux comptes.

Évitez notamment d'utiliser `"Resource": "*"` dans une instruction de politique qui autorise les opérations d'API suivantes. Ces opérations peuvent être appelées sur des clés KMS dans d'autres Comptes AWS.

- DescribeKey
- GetKeyRotationStatus
- Opérations cryptographiques (chiffrer, déchiffrer,,,, GenerateDataKey, GenerateDataKeyPair,, GenerateDataKeyWithoutPlaintext, GenerateDataKeyPairWithoutPlaintext, signer GetPublicKeyReEncrypt, vérifier)
- CreateGrant, ListGrants, ListRetirableGrants, RetireGrant, RevokeGrant

Quand utiliser « Resource » : « * »

Dans une politique IAM, utilisez un caractère générique dans l'élément `Resource` uniquement pour les autorisations qui le requièrent. Seules les autorisations suivantes nécessitent l'élément `"Resource": "*"`.

- km : CreateKey
- km : GenerateRandom
- km : ListAliases
- km : ListKeys

- Autorisations pour les magasins de clés personnalisés, tels que kms : CreateCustomKeyStore et kms : ConnectCustomKeyStore.

> **Note**
>
> Les autorisations pour les opérations d'alias (kms : CreateAlias, kms : UpdateAlias, kms : DeleteAlias) doivent être associées à l'alias et à la clé KMS. Vous pouvez utiliser `"Resource": "*"` dans une politique IAM pour représenter les alias et les clés KMS, ou spécifier les alias et les clés KMS dans l'élément `Resource`. Pour obtenir des exemples, consultez Contrôle de l'accès aux alias.

Les exemples de cette rubrique fournissent plus d'informations et de conseils sur la conception de politiques IAM pour les clés KMS. Pour connaître les meilleures pratiques en matière d'IAM pour toutes les AWS ressources, consultez la section Bonnes pratiques de sécurité en matière d'IAM dans le guide de l'utilisateur d'IAM.

Spécification de clés KMS dans les instructions de politique IAM

Vous pouvez utiliser une politique IAM pour permettre à un principal d'utiliser ou de gérer des clés KMS. Les clés KMS sont spécifiées dans l'élément `Resource` de l'instruction de politique.

- Pour spécifier une clé KMS dans une instruction de politique IAM, vous devez utiliser son ARN de clé. Vous ne pouvez pas utiliser un ID de clé, un nom d'alias ou un ARN d'alias pour identifier une clé KMS dans une instruction de politique IAM.

  Par exemple : « `Resource": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab` »

  Pour contrôler l'accès à une clé KMS en fonction de ses alias, utilisez les clés de ResourceAliases condition kms : RequestAlias ou kms :. Pour plus de détails, consultez ABAC pour AWS KMS.

  Utilisez un alias ARN comme ressource uniquement dans une déclaration de politique qui contrôle l'accès aux opérations d'alias, telles que CreateAliasUpdateAlias, ou DeleteAlias. Pour plus de détails, consultez Contrôle de l'accès aux alias.
- Pour spécifier plusieurs clés KMS dans le compte et la région, utilisez des caractères génériques (*) dans les positions Region (Région) ou Resource ID (ID de ressource) de l'ARN clé.

Par exemple, pour spécifier toutes les clés KMS dans la région USA Ouest (Oregon) d'un compte, utilisez « `Resource": "arn:aws:kms:us-west-2:111122223333:key/*` ». Pour spécifier toutes les clés KMS dans toutes les régions du compte, utilisez « `Resource": "arn:aws:kms:*:111122223333:key/*` ».

- Pour représenter toutes les clés KMS, utilisez un caractère générique seul ("*"). Utilisez ce format pour les opérations qui n'utilisent aucune clé KMS particulière CreateKey, à savoir GenerateRandomListAliases, et ListKeys.

Lorsque vous rédigez vos instructions de politique, il s'agit d'une bonne pratique pour spécifier uniquement les clés KMS que le principal doit utiliser, plutôt que de leur donner accès à toutes les clés KMS.

Par exemple, la déclaration de politique IAM suivante permet au principal d'appeler les DescribeKeyopérations de déchiffrement uniquement sur les clés KMS répertoriées dans l'`Resource`élément de la déclaration de politique. GenerateDataKey La spécification des clés KMS par ARN de clé, qui est une bonne pratique, garantit que les autorisations sont limitées uniquement aux clés KMS spécifiées.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:DescribeKey",
      "kms:GenerateDataKey",
      "kms:Decrypt"
    ],
    "Resource": [
      "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    ]
  }
}
```

Pour appliquer l'autorisation à toutes les clés KMS d'une entité sécurisée donnée Compte AWS, vous pouvez utiliser des caractères génériques (*) dans la région et les positions des identifiants clés. Par exemple, l'instruction de politique suivante permet au principal d'appeler les opérations spécifiées sur toutes les clés KMS dans deux exemples de comptes de confiance.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:DescribeKey",
      "kms:GenerateDataKey",
      "kms:GenerateDataKeyPair"
    ],
    "Resource": [
      "arn:aws:kms:*:111122223333:key/*",
      "arn:aws:kms:*:444455556666:key/*"
    ]
  }
}
```

Vous pouvez également utiliser un caractère générique (`"*"`) seul dans l'élément `Resource`. Comme il permet l'accès à toutes les clés KMS que le compte a l'autorisation d'utiliser, il est recommandé principalement pour les opérations sans clé KMS particulière et pour les instructions `Deny`. Vous pouvez également l'utiliser dans des instructions de politique qui autorisent uniquement des opérations moins sensibles en lecture seule. Pour déterminer si une AWS KMS opération implique une clé KMS particulière, recherchez la valeur de la clé KMS dans la colonne Ressources du tableau dethe section called “Référence des autorisations ”.

Par exemple, l'instruction de politique suivante utilise un effet `Deny` pour interdire aux principaux d'utiliser les opérations spécifiées sur une clé KMS. Elle utilise un caractère générique dans l'élément `Resource` pour représenter toutes les clés KMS.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
      "kms:CreateKey",
      "kms:PutKeyPolicy",
      "kms:CreateGrant",
      "kms:ScheduleKeyDeletion"
    ],
    "Resource": "*"
  }
}
```

L'instruction de politique suivante utilise un caractère générique seul pour représenter toutes les clés KMS. Mais il n'autorise que les opérations moins sensibles en lecture seule et les opérations qui ne s'appliquent pas à une clé KMS particulière.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:CreateKey",
      "kms:ListKeys",
      "kms:ListAliases",
      "kms:ListResourceTags"
    ],
    "Resource": "*"
  }
}
```

Exemples de politique IAM

Dans cette section, vous trouverez des exemples de politiques IAM qui accordent des autorisations pour diverses actions AWS KMS .

> **Important**
>
> Certaines des autorisations figurant dans les politiques suivantes sont autorisées uniquement lorsque la politique de clé de la clé KMS les autorise également. Pour de plus amples informations, veuillez consulter Référence des autorisations .

Pour obtenir de l'aide sur la rédaction et la mise en forme d'un document de politique JSON, veuillez consulter Référence de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples

- Autoriser un utilisateur à afficher les clés KMS dans la AWS KMS console
- Autoriser un utilisateur à créer des clés KMS
- Permettre à un utilisateur de chiffrer et de déchiffrer avec n'importe quelle clé KMS dans un domaine spécifique Compte AWS

- Autoriser un utilisateur à chiffrer et déchiffrer avec n'importe quelle clé KMS dans une région et une région spécifiques Compte AWS
- Autoriser un utilisateur à chiffrer et déchiffrer des données avec des clés KMS spécifiques
- Empêcher un utilisateur de désactiver et de supprimer des clés KMS

Autoriser un utilisateur à afficher les clés KMS dans la AWS KMS console

La politique IAM suivante permet aux utilisateurs d'accéder à la console en lecture seule. AWS KMS Les utilisateurs disposant de ces autorisations peuvent voir toutes les clés KMS qu'ils Compte AWS contiennent, mais ils ne peuvent pas créer ou modifier de clés KMS.

Pour afficher les clés KMS sur les pages des clés gérées par le client Clés gérées par AWSet sur les pages des clés gérées par le clientListKeys, les principaux ont besoin des GetResources autorisations kms : ListAliases, kms : et tag :, même si les clés ne comportent pas de balises ni d'alias. Les autorisations restantes, en particulier kms : DescribeKey, sont requises pour afficher les colonnes facultatives du tableau des clés KMS et les données sur les pages détaillées des clés KMS. Les ListRoles autorisations iam : ListUsers et iam : sont requises pour afficher la politique clé dans l'affichage par défaut sans erreur. Pour consulter les données sur la page des magasins de clés personnalisés et les détails sur les clés KMS dans les magasins de clés personnalisés, les principaux ont également besoin de DescribeCustomKeyStores l'autorisation kms :.

Si vous limitez l'accès de la console d'un utilisateur à des clés KMS particulières, la console affiche une erreur pour chaque clé KMS qui n'est pas visible.

Cette politique inclut deux instructions de politique. L'élément `Resource` dans la première instruction de politique accorde les autorisations spécifiées sur toutes les clés KMS dans toutes les régions du Compte AWS d'exemple. Les utilisateurs de la console n'ont pas besoin d'un accès supplémentaire, car la console AWS KMS affiche uniquement les clés KMS dans le compte du principal. Cela est vrai même s'ils sont autorisés à afficher les clés KMS dans d'autres langues Comptes AWS. Les autorisations restantes AWS KMS et IAM nécessitent un `"Resource": "*"` élément car elles ne s'appliquent à aucune clé KMS en particulier.

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

```
        "kms:GetPublicKey",
        "kms:GetKeyRotationStatus",
        "kms:GetKeyPolicy",
        "kms:DescribeKey",
        "kms:ListKeyPolicies",
        "kms:ListResourceTags",
        "tag:GetResources"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    },
    {
      "Sid": "ReadOnlyAccessForOperationsWithNoKMSKey",
      "Effect": "Allow",
      "Action": [
        "kms:ListKeys",
        "kms:ListAliases",
        "iam:ListRoles",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

Autoriser un utilisateur à créer des clés KMS

La politique IAM suivante permet à un utilisateur de créer tous les types de clés KMS. La valeur de l'`Resource`élément est * due au fait que l'`CreateKey`opération n'utilise aucune AWS KMS ressource particulière (clés KMS ou alias).

Pour restreindre l'utilisateur à certains types de clés KMS, utilisez les clés de KeyOrigin condition kms : KeyUsage, kms : et kms :. KeySpec

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "kms:CreateKey",
    "Resource": "*"
  }
}
```

Les principaux qui créent des clés peuvent avoir besoin de certaines autorisations associées.

- kms : PutKeyPolicy — Les principaux `kms:CreateKey` autorisés peuvent définir la politique de clé initiale pour la clé KMS. Cependant, l'`CreateKey`appelant doit disposer de l'PutKeyPolicyautorisation kms :, qui lui permet de modifier la politique des clés KMS, ou il doit spécifier le `BypassPolicyLockoutSafetyCheck` paramètre de`CreateKey`, ce qui n'est pas recommandé. L'appelant `CreateKey` peut obtenir l'autorisation `kms:PutKeyPolicy` pour la clé KMS depuis une politique IAM, ou il peut inclure cette autorisation dans la politique de clé de la clé KMS qu'il crée.
- kms : TagResource — Pour ajouter des balises à la clé KMS pendant l'`CreateKey`opération, l'`CreateKey`appelant doit disposer de l'TagResourceautorisation kms : dans une politique IAM. L'inclusion de cette autorisation dans la politique de clé de la nouvelle clé KMS ne suffit pas. Cependant, si l'appelant `CreateKey` inclut `kms:TagResource` dans la politique de clé initiale, il peut ajouter des balises dans un appel séparé après la création de la clé KMS.
- kms : CreateAlias — Les principaux qui créent une clé KMS dans la AWS KMS console doivent disposer de l'CreateAliasautorisation kms : sur la clé KMS et sur l'alias. (La console effectue deux appels ; un à `CreateKey` et un à `CreateAlias`). Vous devez fournir l'autorisation d'alias dans une politique IAM. Vous pouvez fournir l'autorisation de clé KMS dans une politique de clé ou une politique IAM. Pour plus de détails, consultez Contrôle de l'accès aux alias.

En outre`kms:CreateKey`, la politique IAM suivante fournit des `kms:TagResource` autorisations sur toutes les clés KMS du compte Compte AWS et des `kms:CreateAlias` autorisations sur tous les alias du compte. Elle inclut également certaines autorisations utiles en lecture seule qui peuvent être fournies uniquement dans une politique IAM.

Cette politique IAM n'inclut pas l'autorisation `kms:PutKeyPolicy` ou toute autre autorisation pouvant être définie dans une politique de clé. La définition de ces autorisations dans la politique de clé, où elles s'appliquent exclusivement à une clé KMS, est une bonne pratique.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPermissionsForParticularKMSKeys",
      "Effect": "Allow",
      "Action": "kms:TagResource",
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    },
    {
      "Sid": "IAMPermissionsForParticularAliases",
```

```
      "Effect": "Allow",
      "Action": "kms:CreateAlias",
      "Resource": "arn:aws:kms:*:111122223333:alias/*"
    },
    {
      "Sid": "IAMPermissionsForAllKMSKeys",
      "Effect": "Allow",
      "Action": [
        "kms:CreateKey",
        "kms:ListKeys",
        "kms:ListAliases"
      ],
      "Resource": "*"
    }
  ]
}
```

Permettre à un utilisateur de chiffrer et de déchiffrer avec n'importe quelle clé KMS dans un domaine spécifique Compte AWS

La politique IAM suivante permet à un utilisateur de chiffrer et de déchiffrer des données avec n'importe quelle clé KMS dans le 111122223333. Compte AWS

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt"
    ],
    "Resource": "arn:aws:kms:*:111122223333:key/*"
  }
}
```

Autoriser un utilisateur à chiffrer et déchiffrer avec n'importe quelle clé KMS dans une région et une région spécifiques Compte AWS

La politique IAM suivante permet à un utilisateur de chiffrer et de déchiffrer des données avec n'importe quelle clé KMS Compte AWS 111122223333 dans la région de l'ouest des États-Unis (Oregon).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt"
    ],
    "Resource": [
      "arn:aws:kms:us-west-2:111122223333:key/*"
    ]
  }
}
```

Autoriser un utilisateur à chiffrer et déchiffrer des données avec des clés KMS spécifiques

La politique IAM suivante permet à un utilisateur de chiffrer et de déchiffrer des données avec les deux clés KMS spécifiées dans l'élément `Resource`. Lorsque vous spécifiez une clé KMS dans une instruction de politique IAM, vous devez utiliser l'ARN de clé de la clé KMS.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt"
    ],
    "Resource": [
      "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    ]
  }
}
```

Empêcher un utilisateur de désactiver et de supprimer des clés KMS

La politique IAM suivante empêche un utilisateur de désactiver et de supprimer des clés KMS, même si une autre politique IAM ou une politique de clé accorde ces autorisations. Une politique qui refuse explicitement des autorisations se substitue à toutes les autres politiques, même à celles

qui accordent explicitement les mêmes autorisations. Pour de plus amples informations, veuillez consulter AWS KMS Permissions de résolution des.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
      "kms:DisableKey",
      "kms:ScheduleKeyDeletion"
    ],
    "Resource": "*"
  }
}
```

Politiques de contrôle des ressources dans AWS KMS

Les politiques de contrôle des ressources (RCPs) sont un type de politique d'entreprise que vous pouvez utiliser pour appliquer des contrôles préventifs sur AWS les ressources de votre organisation. RCPs vous aider à restreindre de manière centralisée l'accès externe à vos AWS ressources à grande échelle. RCPs compléter les politiques de contrôle des services (SCPs). Bien que cela SCPs puisse être utilisé pour définir de manière centralisée les autorisations maximales sur les rôles et les utilisateurs IAM de votre organisation, il RCPs peut être utilisé pour définir de manière centralisée les autorisations maximales sur les AWS ressources de votre organisation.

Vous pouvez l'utiliser RCPs pour gérer les autorisations relatives aux clés KMS gérées par le client dans votre organisation. RCPs ne suffisent pas à elles seules à accorder des autorisations aux clés gérées par vos clients. Aucune autorisation n'est accordée par un RCP. Un RCP définit un garde-fou en matière d'autorisations, ou fixe des limites, aux actions que les identités peuvent effectuer sur les ressources des comptes concernés. L'administrateur doit toujours associer des politiques basées sur l'identité aux rôles ou aux utilisateurs IAM, ou des politiques clés pour réellement accorder des autorisations.

> **Note**
>
> Les politiques de contrôle des ressources de votre organisation ne s'appliquent pas à Clés gérées par AWS.

Clés gérées par AWS sont créés, gérés et utilisés en votre nom par un AWS service, vous ne pouvez ni modifier ni gérer ses autorisations.

En savoir plus

- Pour des informations plus générales RCPs, voir les politiques de contrôle des ressources dans le guide de AWS Organizations l'utilisateur.
- Pour plus de détails sur la façon de définir RCPs, y compris des exemples, consultez la section Syntaxe RCP dans le guide de AWS Organizations l'utilisateur.

L'exemple suivant montre comment utiliser un RCP pour empêcher les principaux externes d'accéder aux clés gérées par le client dans votre organisation. Cette politique n'est qu'un exemple, et vous devez l'adapter à vos besoins commerciaux et de sécurité uniques. Par exemple, vous souhaiterez peut-être personnaliser votre politique pour autoriser l'accès à vos partenaires commerciaux. Pour plus de détails, consultez le référentiel d'exemples de politiques de périmètre des données.

Note

L'`kms:RetireGrant`autorisation n'est pas effective dans un RCP, même si l'`Action`élément indique un astérisque (*) comme caractère générique.
Pour plus d'informations sur le mode de détermination `kms:RetireGrant` de l'autorisation, consultezRetrait et révocation d'octrois.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RCPEnforceIdentityPerimeter",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "kms:*",
            "Resource": "*",
            "Condition": {
                "StringNotEqualsIfExists": {
                    "aws:PrincipalOrgID": "my-org-id"
                },
                "Bool": {
```

```
                    "aws:PrincipalIsAWSService": "false"
                }
            }
        }
    ]
}
```

L'exemple de RCP suivant exige que les responsables du AWS service puissent accéder à vos clés KMS gérées par le client uniquement lorsque la demande provient de votre organisation. Cette politique applique le contrôle uniquement aux demandes `aws:SourceAccount` présentes. Cela garantit que les intégrations de services qui ne nécessitent pas l'utilisation de `aws:SourceAccount` ne sont pas affectées. Si elle `aws:SourceAccount` est présente dans le contexte de la demande, la `Null` condition est évaluée à`true`, ce qui entraîne l'application de la `aws:SourceOrgID` clé.

Pour plus d'informations sur le problème des adjoints confus, voir Le problème des adjoints confus dans le guide de l'utilisateur d'IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RCPEnforceConfusedDeputyProtection",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "kms:*",
            "Resource": "*",
            "Condition": {
                "StringNotEqualsIfExists": {
                    "aws:SourceOrgID": "my-org-id"
                },
                "Bool": {
                    "aws:PrincipalIsAWSService": "true"
                },
                "Null": {
                    "aws:SourceAccount": "false"
                }
            }
        }
    ]
}
```

Subventions en AWS KMS

Un octroi est un instrument de politique qui permet aux principaux AWS d'utiliser des clés KMS dans les opérations de chiffrement. Il peut également leur permettre d'afficher une clé KMS (`DescribeKey`), mais aussi de créer et de gérer des octrois. Lorsque vous autorisez l'accès à une clé KMS, les octrois sont pris en compte avec des politiques de clé et des politiques IAM. Les octrois sont souvent utilisés pour des autorisations temporaires, car vous pouvez en créer un, utiliser ses autorisations et les supprimer sans modifier vos politiques de clé ou IAM.

Les subventions sont couramment utilisées par AWS les services qui s'intègrent AWS KMS pour chiffrer vos données au repos. Le service crée un octroi au nom d'un utilisateur du compte, utilise ses autorisations et retire l'octroi dès que sa tâche est terminée. Pour plus de détails sur la manière dont les AWS services utilisent les subventions, consultez la rubrique Chiffrement au repos du guide de l'utilisateur ou du guide du développeur du service.

Les octrois sont un mécanisme de contrôle d'accès très souple et utile. Lorsque vous créez un octroi pour une clé KMS, celui-ci permet au principal bénéficiaire d'appeler les opérations d'octroi spécifiées sur la clé KMS, à condition que toutes les conditions spécifiées dans l'octroi soient remplies.

- Chaque octroi permet d'accéder à exactement une clé KMS. Vous pouvez créer un octroi pour une clé KMS dans un autre Compte AWS.
- Un octroi peut autoriser l'accès à une clé KMS, mais pas le lui refuser.
- Chaque octroi a un principal bénéficiaire. Le principal du bénéficiaire peut représenter une ou plusieurs identités au même Compte AWS titre que la clé KMS ou dans un compte différent.
- Un octroi peut uniquement permettre des opérations d'octroi. Les opérations d'octroi doivent être prises en charge par la clé KMS de l'octroi. Si vous spécifiez une opération non prise en charge, la CreateGrantdemande échoue avec une `ValidationError` exception.
- Le principal bénéficiaire peut utiliser les autorisations que l'octroi lui donne sans spécifier l'octroi, comme il le ferait si les autorisations provenaient d'une politique de clé ou d'une politique IAM. Cependant, étant donné que l' AWS KMS API suit un modèle de cohérence final, lorsque vous créez, retirez ou révoquez une subvention, il peut y avoir un bref délai avant que la modification ne soit disponible dans AWS KMS son intégralité. Pour utiliser immédiatement les autorisations dans un octroi, utilisez un jeton d'octroi.
- Un principal autorisé peut supprimer l'octroi (le retirer ou le révoquer). La suppression d'un octroi élimine toutes les autorisations qu'il accorde. Vous n'avez pas besoin de déterminer les politiques à ajouter ou à supprimer pour annuler l'octroi.

- AWS KMS limite le nombre d'autorisations sur chaque clé KMS. Pour plus de détails, consultez Octrois par clé KMS : 50 000.

Soyez prudent lorsque vous créez des octrois et lorsque vous autorisez d'autres personnes à en créer. L'autorisation de créer des subventions a des implications en matière de sécurité, tout comme l'PutKeyPolicyautorisation de définir des politiques en termes de kilomètres.

- Les utilisateurs autorisés à créer des autorisations pour une clé KMS (`kms:CreateGrant`) peuvent utiliser une autorisation pour autoriser les utilisateurs et les rôles, y compris les AWS services, à utiliser la clé KMS. Les principes peuvent être des identités propres Compte AWS ou des identités appartenant à un autre compte ou à une autre organisation.
- Les subventions ne peuvent autoriser qu'un sous-ensemble d' AWS KMS opérations. Vous pouvez utiliser des octrois pour autoriser les principaux à afficher la clé KMS, à l'utiliser dans les opérations de chiffrement, mais aussi à créer et à retirer des octrois. Pour plus d'informations, veuillez consulter .Opérations d'octroi. Vous pouvez également utiliser des contraintes d'octroi pour limiter les autorisations dans un octroi pour une clé de chiffrement symétrique.
- Les principaux peuvent obtenir l'autorisation de créer des octrois à partir d'une politique de clé ou d'une politique IAM. Les directeurs qui obtiennent `kms:CreateGrant` l'autorisation d'une politique peut créer des subventions pour tout opération d'octroi sur la clé KMS. Ces mandants ne sont pas tenus d'avoir l'autorisation qu'ils accordent sur la clé. Lorsque vous accordez l'autorisation `kms:CreateGrant` dans une politique, vous pouvez utiliser des conditions de politique pour limiter cette autorisation.
- Les principaux peuvent également obtenir l'autorisation de créer des octrois à partir d'un octroi. Ces principaux peuvent uniquement déléguer les autorisations qui leur ont été accordées, même s'ils disposent d'autres autorisations en vertu d'une politique. Pour plus de détails, consultez Octroi CreateGrant d'autorisation.

Concepts d'octroi

Pour utiliser efficacement les octrois, vous devez comprendre les termes et les concepts utilisés par AWS KMS .

Contrainte d'octroi

Condition qui limite les autorisations dans l'octroi. Actuellement, AWS KMS prend en charge les contraintes d'octroi basées sur le contexte de chiffrement dans la demande d'opération cryptographique. Pour plus de détails, consultez Utilisation des contraintes d'octroi.

ID d'octroi

Identifiant unique d'un octroi pour une clé KMS. Vous pouvez utiliser un identifiant de subvention, ainsi qu'un identifiant clé, pour identifier une autorisation dans une RevokeGrantdemande RetireGrantou.

Opérations d'octroi

Les AWS KMS opérations que vous pouvez autoriser dans le cadre d'une subvention. Si vous spécifiez d'autres opérations, la CreateGrantdemande échoue avec une `ValidationError` exception. Ce sont aussi les opérations qui acceptent un jeton d'octroi. Pour de plus amples informations sur ces autorisations, veuillez consulter la AWS KMS autorisations.

Ces opérations d'octroi représentent effectivement l'autorisation d'utiliser l'opération. Par conséquent, pour l'opération `ReEncrypt`, vous pouvez spécifier `ReEncryptFrom`, `ReEncryptTo` ou les deux `ReEncrypt*`.

Les opérations d'octroi sont les suivantes :

- Opérations cryptographiques
  - Decrypt (Déchiffrer)
  - DeriveSharedSecret
  - Encrypt (Chiffrer)
  - GenerateDataKey
  - GenerateDataKeyPair
  - GenerateDataKeyPairWithoutPlaintext
  - GenerateDataKeyWithoutPlaintext
  - GenerateMac
  - ReEncryptFrom
  - ReEncryptTo
  - Sign (Signer)
  - Verify (Vérifier)
  - VerifyMac

- Autres opérations
  - CreateGrant
  - DescribeKey
  - GetPublicKey
  - RetireGrant

  Les opérations d'octroi que vous autorisez doivent être prises en charge par la clé KMS de l'octroi. Si vous spécifiez une opération non prise en charge, la CreateGrantdemande échoue avec une `ValidationError` exception. Par exemple, les octrois pour les clés KMS de chiffrement symétrique ne peuvent pas autoriser les opérations Sign (Signer), Verify (Vérifier), `GenerateMac` ou `VerifyMac`. Les octrois pour les clés KMS asymétriques ne peuvent autoriser aucune opération générant des clés de données ou des paires de clés de données.

Jeton d'octroi

L' AWS KMS API suit un modèle de cohérence éventuel. Lorsque vous créez un octroi, il se peut qu'il y ait un bref délai avant que le changement ne soit disponible via AWS KMS. La propagation de la modification dans l'ensemble du système prend généralement moins de quelques secondes, mais dans certains cas, cela peut prendre plusieurs minutes. Si vous essayez d'utiliser un octroi avant d'être propagé complètement sur le système, vous pouvez obtenir un message d'accès refusé. Un jeton d'octroi vous permet de faire référence à l'octroi et d'utiliser les autorisations d'octroi immédiatement.

Un jeton d'octroi est une chaîne unique, non secrète, de longueur variable et codée en base64 qui représente un octroi. Vous pouvez utiliser le jeton d'octroi pour identifier l'octroi dans n'importe quelle opération d'octroi. Cependant, comme la valeur du jeton est un résumé de hachage, elle ne révèle aucun détail sur l'octroi.

Un jeton d'octroi est conçu pour être utilisé uniquement jusqu'à ce qu'il se propage complètement sur AWS KMS. Après cela, le principal bénéficiaire peut utiliser l'autorisation dans l'octroi sans fournir de jeton d'octroi ou toute autre preuve de l'octroi. Vous pouvez utiliser un jeton de subvention à tout moment, mais une fois que l'autorisation est finalement cohérente, c'est AWS KMS l'autorisation qui détermine les autorisations, et non le jeton de subvention.

Par exemple, la commande suivante appelle l'GenerateDataKeyopération. Elle utilise un jeton d'octroi pour représenter l'octroi qui donne à l'appelant (le principal bénéficiaire) l'autorisation d'appeler `GenerateDataKey` sur la clé KMS spécifiée.

```
$ aws kms generate-data-key \
```

```
        --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
        --key-spec AES_256 \
        --grant-token $token
```

Vous pouvez également utiliser le jeton d'octroi pour identifier un octroi dans les opérations qui gèrent les octrois. Par exemple, le directeur sortant peut utiliser un jeton de subvention lors d'un appel à l'RetireGrantopération.

```
$ aws kms retire-grant \
        --grant-token $token
```

`CreateGrant` est la seule opération qui renvoie un jeton d'octroi. Vous ne pouvez pas obtenir de jeton d'autorisation à partir d'une autre AWS KMS opération ou du CloudTrail journal des événements associés à l' CreateGrant opération. Les ListRetirableGrantsopérations ListGrantset renvoient l'ID de subvention, mais pas un jeton de subvention.

Pour plus de détails, consultez Utilisation d'un jeton d'octroi.

Principal bénéficiaire

Identité qui obtient les autorisations spécifiées dans l'octroi. Chaque octroi n'a qu'un seul principal bénéficiaire, mais ce dernier peut représenter plusieurs identités.

Le principal bénéficiaire peut être n'importe quel AWS principal, y compris un Compte AWS (root), un utilisateur IAM, un rôle IAM, un rôle ou un utilisateur fédéré, ou un utilisateur ayant un rôle assumé. Le principal bénéficiaire peut se trouver dans le même compte que la clé KMS ou un autre compte. Toutefois, le principal bénéficiaire ne peut pas être un principal de service, un groupe IAM ou une organisation AWS.

> **Note**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Retirer (un octroi)

Résilie un octroi. Vous retirez un octroi lorsque vous avez terminé d'utiliser les autorisations.

La révocation et le retrait d'un octroi suppriment l'octroi. Toutefois, le retrait est effectué par un principal spécifié dans l'octroi. La révocation est généralement effectuée par un administrateur de clé. Pour plus de détails, consultez Retrait et révocation d'octrois.

Principal de retrait

Un principal qui peut retirer un octroi. Vous pouvez spécifier un principal de retrait dans un octroi, mais ce n'est pas obligatoire. Le mandant sortant peut être n'importe quel AWS mandant, y compris les utilisateurs IAM Comptes AWS, les rôles IAM, les utilisateurs fédérés et les utilisateurs des rôles assumés. Le principal de retrait peut se trouver dans le même compte que la clé KMS ou un autre compte.

> **Note**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Outre le remboursement du capital spécifié dans la subvention, une subvention peut être retirée par le fournisseur Compte AWS dans lequel la subvention a été créée. Si l'octroi autorise l'opération `RetireGrant`, le principal bénéficiaire peut retirer l'octroi. En outre, le Compte AWS directeur sortant peut déléguer l'autorisation de retirer une subvention à un directeur IAM du même nom. Compte AWS Compte AWS Pour plus de détails, consultez Retrait et révocation d'octrois.

Révoquer (un octroi)

Résilie un octroi. Vous révoquez un octroi pour refuser activement les autorisations que l'octroi autorise.

La révocation et le retrait d'un octroi suppriment l'octroi. Toutefois, le retrait est effectué par un principal spécifié dans l'octroi. La révocation est généralement effectuée par un administrateur de clé. Pour plus de détails, consultez Retrait et révocation d'octrois.

Cohérence éventuelle (pour les octrois)

L' AWS KMS API suit un modèle de cohérence éventuel. Lorsque vous créez, retirez ou révoquez un octroi, il se peut qu'il y ait un bref délai avant que le changement ne soit disponible via AWS

KMS. La propagation de la modification dans l'ensemble du système prend généralement moins de quelques secondes, mais dans certains cas, cela peut prendre plusieurs minutes.

Vous pouvez prendre connaissance de ce bref délai si vous obtenez des erreurs inattendues. Par exemple, si vous essayez de gérer une nouvelle subvention ou d'utiliser les autorisations d'une nouvelle autorisation avant que l'autorisation ne soit connue dans son intégralité AWS KMS, vous risquez de recevoir un message d'erreur de refus d'accès. Si vous retirez ou révoquez un octroi, le principal bénéficiaire peut toujours utiliser ses autorisations pendant une courte période jusqu'à ce que l'octroi soit complètement supprimé. La stratégie classique consiste à réessayer la demande, et certaines AWS SDKs incluent une logique d'annulation automatique et de nouvelle tentative.

AWS KMS possède des fonctionnalités pour atténuer ce bref délai.

- Pour utiliser immédiatement les autorisations dans un nouvel octroi, utilisez un jeton d'octroi. Vous pouvez utiliser un jeton d'octroi pour faire référence à un octroi dans n'importe quelle opération d'octroi. Pour obtenir des instructions, consultez Utilisation d'un jeton d'octroi.
- L'CreateGrantopération possède un `Name` paramètre qui empêche les nouvelles tentatives de créer des autorisations dupliquées.

> **Note**
>
> Les jetons d'octroi remplacent la validité de l'octroi jusqu'à ce que tous les points de terminaison du service aient été mis à jour avec le nouvel état de l'octroi. Dans la plupart des cas, une cohérence éventuelle sera obtenue dans les cinq minutes.

Pour plus d'informations, consultez la rubrique relative à la cohérence à terme AWS KMS.

Bonnes pratiques en matière de AWS KMS subventions

AWS KMS recommande les meilleures pratiques suivantes lors de la création, de l'utilisation et de la gestion des subventions.

- Limitez les autorisations de l'octroi aux autorisations requises par le principal bénéficiaire. Utilisez le principe d'accès le moins privilégié.
- Utilisez un principal bénéficiaire spécifique, tel qu'un rôle IAM, et donnez au principal l'autorisation d'utiliser uniquement les opérations API dont il a besoin.

- Utilisez le contexte de chiffrement de contraintes d'octroi pour garantir que les appelants utilisent la clé KMS aux fins prévues. Pour en savoir plus sur l'utilisation du contexte de chiffrement dans une demande de sécurisation de vos données, consultez Comment protéger l'intégrité de vos données chiffrées en utilisant AWS Key Management Service et EncryptionContext dans le blog sur la AWS sécurité.

  > **Tip**
  >
  > Utilisez la contrainte de EncryptionContextEqualsubvention dans la mesure du possible. La contrainte de EncryptionContextSubsetsubvention est plus difficile à utiliser correctement. Si vous devez l'utiliser, lisez attentivement la documentation et testez la contrainte d'octroi pour vous assurer qu'elle fonctionne comme prévu.

- Supprimer les octrois en double. Les octrois en double ont les mêmes ARN de clé, actions d'API, principal bénéficiaire, contexte de chiffrement et nom. Si vous retirez ou révoquez l'octroi initial, mais que vous laissez les doublons, les doublons restants constituent une escalade involontaire de privilèges. Pour éviter de dupliquer les octrois lors de la relance d'une demande `CreateGrant`, utilisez le paramètre `Name`. Pour détecter les autorisations dupliquées, utilisez l'ListGrantsopération. Si vous créez accidentellement un octroi en double, retirez-le ou révoquez-le dès que possible.

  > **Note**
  >
  > Les octrois pour les clés gérées par AWS peuvent ressembler à des doublons, mais ont des principaux bénéficiaires différents.
  > Le champ `GranteePrincipal` de la réponse `ListGrants` contient habituellement le bénéficiaire principal. Toutefois, lorsque le principal bénéficiaire de la subvention est un AWS service, le `GranteePrincipal` champ contient le principal du service, qui peut représenter plusieurs principaux bénéficiaires différents.

- N'oubliez pas que les octrois n'expirent pas automatiquement. Retirez ou révoquez l'octroi dès que l'autorisation n'est plus nécessaire. Les octrois qui ne sont pas supprimés peuvent créer un risque de sécurité pour les ressources chiffrées.

Contrôle de l'accès aux octrois

Vous pouvez contrôler l'accès aux opérations qui créent et gèrent des octrois dans les politiques de clés, les politiques IAM et les octrois. Les principaux qui obtiennent l'autorisation `CreateGrant` d'un octroi ont des autorisations d'octroi plus limitées.

Opération API	politique de clé ou politique IAM	Grant (Octroi)
CreateGrant	✓	✓
ListGrants	✓	-
ListRetirableGrants	✓	-
Retirer des octrois	(Limité. Voir Retrait et révocation d'octrois)	✓
RevokeGrant	✓	-

Lorsque vous utilisez une politique clé ou une politique IAM pour contrôler l'accès aux opérations qui créent et gèrent des autorisations, vous pouvez utiliser une ou plusieurs des conditions de politique suivantes pour limiter l'autorisation. AWS KMS prend en charge toutes les clés de condition relatives aux subventions suivantes. Pour plus d'informations et d'exemples, veuillez consulter AWS KMS clés de condition.

km : GrantConstraintType

Permet aux principaux de créer un octroi uniquement lorsque l'octroi inclut la contrainte d'octroispécifiée.

km : GrantIsFor AWSResource

Permet aux principaux d'appeler `CreateGrantListGrants`, ou `RevokeGrant` uniquement lorsqu'un AWS service intégré AWS KMS envoie la demande au nom du principal.

km : GrantOperations

Autorise les principaux à créer un octroi, mais limite l'octroi aux opérations spécifiées.

km : GranteePrincipal

Autorise les principaux à créer un octroi uniquement pour le principal bénéficiaire spécifié.

km : RetiringPrincipal

Permet aux principaux de créer un octroi uniquement lorsque l'octroi spécifie un principal de retrait.

Création d'octrois

Avant de créer un octroi, découvrez ses options de personnalisation. Vous pouvez utiliser des contraintes d'octroi pour limiter les autorisations dans l'octroi. En outre, renseignez-vous sur l'autorisation `CreateGrant` d'octroi. Les principaux qui obtiennent l'autorisation de créer des octrois à partir d'un octroi sont limités au niveau des octrois qu'ils peuvent créer.

Rubriques

- Création d'un octroi
- Octroi CreateGrant d'autorisation

Création d'un octroi

Pour créer une subvention, appelez l'CreateGrantopération. Spécifiez une clé KMS, un principal bénéficiaire et une liste des opérations d'octroi autorisées. Vous pouvez également désigner un principal de retrait facultatif. Pour personnaliser l'octroi, utilisez des paramètres `Constraints` facultatifs pour définir les contraintes d'octroi.

Lorsque vous créez, retirez ou révoquez un octroi, il peut y avoir un bref délai, généralement de moins de cinq minutes, avant que la modification ne soit disponible sur AWS KMS. Pour plus d'informations, consultez la rubrique relative à la cohérence à terme (pour les octrois).

Par exemple, la commande `CreateGrant` suivante crée un octroi qui permet aux utilisateurs autorisés à assumer le rôle `keyUserRole` d'appeler l'opération Decrypt (Déchiffrer) sur la clé KMS symétrique spécifiée. L'autorisation utilise le paramètre `RetiringPrincipal` pour désigner un principal qui peut retirer l'autorisation. Elle inclut également une contrainte d'autorisation qui l'autorise uniquement lorsque le contexte de chiffrement de la requête inclut `"Department": "IT"`.

```
$  aws kms create-grant \
```

```
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \
    --operations Decrypt \
    --retiring-principal arn:aws:iam::111122223333:role/adminRole \
    --constraints EncryptionContextSubset={Department=IT}
```

Si votre code relance l'opération `CreateGrant`, ou utilise un kit SDK AWS qui relance automatiquement les demandes, utilisez le paramètre Name (Nom) facultatif pour empêcher la création d'octrois en double. S'il AWS KMS reçoit une `CreateGrant` demande de subvention ayant les mêmes propriétés qu'une subvention existante, y compris le nom, il reconnaît la demande comme une nouvelle tentative et ne crée pas de nouvelle autorisation. Vous pouvez utiliser la valeur `Name` pour identifier l'octroi dans n'importe quelle opération AWS KMS .

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans le nom de l'octroi. Il peut apparaître en texte brut dans CloudTrail les journaux et autres sorties.

```
$ aws kms create-grant \
    --name IT-1234abcd-keyUserRole-decrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \
    --operations Decrypt \
    --retiring-principal arn:aws:iam::111122223333:role/adminRole \
    --constraints EncryptionContextSubset={Department=IT}
```

Pour des exemples de code qui montrent comment créer des subventions dans plusieurs langages de programmation, voirUtilisation `CreateGrant` avec un AWS SDK ou une CLI.

Utilisation des contraintes d'octroi

Les contraintes d'octroi définissent les conditions des autorisations que l'octroi donne au principal bénéficiaire. Les contraintes d'octroi prennent la place de clés de condition dans une politique de clé ou une politique IAM. Chaque valeur de contrainte d'octroi peut inclure jusqu'à 8 paires de contextes de chiffrement. La valeur de contexte de chiffrement dans chaque contrainte d'octroi ne peut pas dépasser 384 caractères.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

AWS KMS prend en charge deux contraintes `EncryptionContextEquals` d'autorisation`EncryptionContextSubset`, qui établissent toutes deux des exigences relatives au contexte de chiffrement dans une demande d'opération cryptographique.

Les contraintes d'octroi du contexte de chiffrement sont conçues pour être utilisées avec des opérations d'octroi qui ont un paramètre de contexte de chiffrement.

- Les contraintes de contexte de chiffrement ne sont valides que dans un octroi pour une clé KMS de chiffrement symétrique. Les opérations de chiffrement avec d'autres clés KMS ne prennent pas en charge un contexte de chiffrement.
- La contrainte de contexte de chiffrement est ignorée pour les opérations `DescribeKey` et `RetireGrant`. `DescribeKey` et `RetireGrant` n'ont pas de paramètre de contexte de chiffrement, mais vous pouvez inclure ces opérations dans un octroi qui a une contrainte de contexte de chiffrement.
- Vous pouvez utiliser une contrainte de contexte de chiffrement dans un octroi pour l'opération `CreateGrant`. La contrainte de contexte de chiffrement nécessite que tous les octrois créés avec l'autorisation `CreateGrant` aient une contrainte de contexte de chiffrement tout aussi stricte ou plus stricte.

AWS KMS prend en charge les contraintes d'octroi de contexte de chiffrement suivantes.

EncryptionContextEquals

Utilisez `EncryptionContextEquals` pour spécifier le contexte de chiffrement exact pour les demandes autorisées.

`EncryptionContextEquals` exige que les paires de contexte de chiffrement de la requête correspondent exactement, y compris au niveau des minuscules/majuscules, aux paires de contexte de chiffrement de la contrainte d'octroi. Les paires peuvent apparaître dans n'importe quel ordre, mais les clés et valeurs dans chaque paire ne peuvent pas varier.

Par exemple, si l'a contrainte d'octroi `EncryptionContextEquals` exige la paire de contextes de chiffrement `"Department": "IT"`, l'octroi autorise les demandes du type spécifié uniquement lorsque le contexte de chiffrement de la requête est exactement `"Department": "IT"`.

EncryptionContextSubset

Utilisez `EncryptionContextSubset` pour exiger que les demandes incluent des paires de contexte de chiffrement particulières.

`EncryptionContextSubset` exige que la demande inclue toutes les paires de contexte de chiffrement de la contrainte d'octroi (une correspondance exacte sensible à la casse), mais la demande peut avoir des paires de contexte de chiffrement supplémentaires. Les paires peuvent apparaître dans n'importe quel ordre, mais les clés et valeurs dans chaque paire ne peuvent pas varier.

Par exemple, si l'a contrainte d'octroi `EncryptionContextSubset` exige la paire de contextes de chiffrement `Department=IT`, l'octroi autorise les demandes du type spécifié uniquement lorsque le contexte de chiffrement de la requête est `"Department": "IT"` ou inclut `"Department": "IT"`, ainsi que d'autres paires de contexte de chiffrement, telles que `"Department": "IT","Purpose": "Test"`.

Pour spécifier une contrainte de contexte de chiffrement dans une autorisation pour une clé KMS de chiffrement symétrique, utilisez le `Constraints` paramètre dans l'CreateGrantopération. L'octroi créé par cette commande accorde aux utilisateurs qui sont autorisés à assumer le rôle `keyUserRole` l'autorisation d'appeler l'opération Decrypt (Déchiffrer). Toutefois, cette autorisation est effective uniquement lorsque le contexte de chiffrement de la demande `Decrypt` est une paire de contextes de chiffrement `"Department": "IT"`.

```
$ aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \
    --operations Decrypt \
    --retiring-principal arn:aws:iam::111122223333:role/adminRole \
    --constraints EncryptionContextEquals={Department=IT}
```

L'octroi obtenu ressemble à ce qui suit. Notez que l'autorisation accordée au rôle `keyUserRole` n'est effective que lorsque la demande `Decrypt` utilise la même paire de contextes de chiffrement que

celle spécifiée dans la contrainte d'octroi. Pour trouver les autorisations sur une clé KMS, utilisez l'ListGrantsopération.

```
$ aws kms list-grants --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "Grants": [
        {
            "Name": "",
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "GrantId":
 "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a",
            "Operations": [
                "Decrypt"
            ],
            "GranteePrincipal": "arn:aws:iam::111122223333:role/keyUserRole",
            "Constraints": {
                "EncryptionContextEquals": {
                    "Department": "IT"
                }
            },
            "CreationDate": 1568565290.0,
            "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "RetiringPrincipal": "arn:aws:iam::111122223333:role/adminRole"
        }
    ]
}
```

Pour satisfaire la contrainte d'octroi `EncryptionContextEquals`, le contexte de chiffrement dans la demande pour l'opération `Decrypt` doit être une paire `"Department": "IT"`. Une demande telle que la suivante émanant du principal bénéficiaire satisferait à la contrainte d'octroi `EncryptionContextEquals`.

```
$ aws kms decrypt \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab\
    --ciphertext-blob fileb://encrypted_msg \
    --encryption-context Department=IT
```

Lorsque la contrainte d'octroi est `EncryptionContextSubset`, les paires de contexte de chiffrement de la demande doivent inclure les paires de contexte de chiffrement dans la contrainte

d'octroi, mais la demande peut également inclure d'autres paires de contexte de chiffrement. La contrainte d'octroi suivante nécessite que l'une des paires de contexte de chiffrement dans la demande soit `"Deparment": "IT"`.

```
"Constraints": {
   "EncryptionContextSubset": {
       "Department": "IT"
   }
}
```

La demande suivante émanant du principal bénéficiaire satisferait à la fois aux contraintes d'octroi `EncryptionContextEqual` et `EncryptionContextSubset` dans cet exemple.

```
$ aws kms decrypt \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --ciphertext-blob fileb://encrypted_msg \
    --encryption-context Department=IT
```

Toutefois, une demande comme celle qui suit émanant du principal bénéficiaire satisferait à la contrainte d'octroi `EncryptionContextSubset`, mais pas à la contrainte d'octroi `EncryptionContextEquals`.

```
$ aws kms decrypt \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --ciphertext-blob fileb://encrypted_msg \
    --encryption-context Department=IT,Purpose=Test
```

AWS les services utilisent souvent des contraintes de contexte de chiffrement dans les autorisations qui leur donnent l'autorisation d'utiliser des clés KMS dans votre Compte AWS. Par exemple, Amazon DynamoDB utilise un octroi comme le suivant pour obtenir l'autorisation d'utiliser la Clé gérée par AWS pour DynamoDB dans votre compte. La contrainte d'octroi `EncryptionContextSubset` de cet octroi rend les autorisations de l'octroi effectives uniquement lorsque le contexte de chiffrement de la demande inclut les paires`"subscriberID": "111122223333"` et `"tableName": "Services"`. Cette contrainte d'octroi signifie que l'octroi autorise DynamoDB à utiliser la clé KMS spécifiée uniquement pour une table particulière de votre Compte AWS.

Pour obtenir ce résultat, exécutez l'ListGrantsopération sur DynamoDB de votre compte. Clé gérée par AWS

```
$ aws kms list-grants --key-id 0987dcba-09fe-87dc-65ba-ab0987654321

{
    "Grants": [
        {
            "Operations": [
                "Decrypt",
                "Encrypt",
                "GenerateDataKey",
                "ReEncryptFrom",
                "ReEncryptTo",
                "RetireGrant",
                "DescribeKey"
            ],
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "Constraints": {
                "EncryptionContextSubset": {
                    "aws:dynamodb:tableName": "Services",
                    "aws:dynamodb:subscriberId": "111122223333"
                }
            },
            "CreationDate": 1518567315.0,
            "KeyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
            "GranteePrincipal": "dynamodb.us-west-2.amazonaws.com",
            "RetiringPrincipal": "dynamodb.us-west-2.amazonaws.com",
            "Name": "8276b9a6-6cf0-46f1-b2f0-7993a7f8c89a",
            "GrantId":
 "1667b97d27cf748cf05b487217dd4179526c949d14fb3903858e25193253fe59"
        }
    ]
}
```

Octroi CreateGrant d'autorisation

Un octroi peut inclure l'autorisation d'appeler l'opération `CreateGrant`. Toutefois, quand un principal bénéficiaire obtient l'autorisation d'appeler `CreateGrant` à partir d'un octroi, plutôt que d'une politique, cette autorisation est limitée.

- Le principal bénéficiaire peut uniquement créer des octrois qui permettent une partie ou la totalité des opérations de l'octroi parent.

- Les contraintes d'octroi dans les octrois qu'elles créent doivent être au moins aussi strictes que celles de l'octroi parent.

Ces limites ne s'appliquent pas aux principaux qui obtiennent l'autorisation `CreateGrant` à partir d'une politique, bien que leurs autorisations puissent être limitées par des conditions de politique.

Par exemple, imaginons un octroi qui autorise le principal bénéficiaire à appeler les opérations `GenerateDataKey`, `Decrypt` et `CreateGrant`. Nous appelons un octroi qui autorise l'autorisation `CreateGrant` d'un octroi parent.

```
# The original grant in a ListGrants response.
{
    "Grants": [
        {
            "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1572216195.0,
            "GrantId":
 "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a",
            "Operations": [
                "GenerateDataKey",
                "Decrypt",
                "CreateGrant
            ]
            "RetiringPrincipal": "arn:aws:iam::111122223333:role/adminRole",
            "Name": "",
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "GranteePrincipal": "arn:aws:iam::111122223333:role/keyUserRole",
            "Constraints": {
                "EncryptionContextSubset": {
                    "Department": "IT"
                }
            },
        }
    ]
}
```

Le principal bénéficiaire, exampleUser, peut utiliser cette autorisation pour créer un octroi qui inclut n'importe quel sous-ensemble des opérations spécifiées dans l'octroi parent, par exemple `CreateGrant` et `Decrypt`. L'octroi enfant ne peut pas inclure d'autres opérations, comme `ScheduleKeyDeletion` ou `ReEncrypt`.

De plus, les contraintes d'octroi des octrois enfants doivent être aussi restrictives, voire plus, que celles de l'octroi parent. Par exemple, l'octroi enfant peut ajouter des paires dans une contrainte `EncryptionContextSubset` de l'octroi parent, mais ne peut pas les supprimer. L'octroi enfant peut modifier une contrainte `EncryptionContextSubset` en contrainte `EncryptionContextEquals`, mais pas l'inverse.

Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Par exemple, le principal bénéficiaire peut utiliser l'autorisation `CreateGrant` qu'il a obtenue de l'octroi parent pour créer l'octroi enfant suivant. Les opérations de l'octroi enfant sont un sous-ensemble des opérations de l'octroi parent et les contraintes d'octroi sont plus restrictives.

```
# The child grant in a ListGrants response.
{
    "Grants": [
        {
            "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1572249600.0,
            "GrantId":
 "fedcba9999c1e2e9876abcde6e9d6c9b6a1987650000abcee009abcdef40183f",
            "Operations": [
                "CreateGrant"
                "Decrypt"
            ]
            "RetiringPrincipal": "arn:aws:iam::111122223333:user/exampleUser",
            "Name": "",
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "GranteePrincipal": "arn:aws:iam::111122223333:user/anotherUser",
            "Constraints": {
                "EncryptionContextEquals": {
                    "Department": "IT"
                }
            },
        }
    ]
}
```

Le principal bénéficiaire de l'octroi enfant ,`anotherUser`, peut utiliser son autorisation `CreateGrant` pour créer des octrois. Cependant, les octrois que `anotherUser` crée doit inclure les opérations dans leur octroi parent ou un sous-ensemble, et les contraintes d'octroi doivent être les mêmes ou plus strictes.

Affichage d'octrois

Pour consulter la subvention, utilisez l'ListGrantsopération. Vous devez spécifier la clé KMS à laquelle les octrois s'appliquent. Vous pouvez également filtrer la liste des octrois par ID d'octroi ou principal bénéficiaire. Pour obtenir plus d'exemples, consultez Utilisation `ListGrants` avec un AWS SDK ou une CLI.

Pour consulter toutes les subventions accordées dans la région Compte AWS et dont le capital est retraité en particulier, utilisez ListRetirableGrants. Les réponses comprennent des détails sur chaque octroi.

> **Note**
>
> Le champ `GranteePrincipal` de la réponse `ListGrants` contient habituellement le bénéficiaire principal. Toutefois, lorsque le principal bénéficiaire de la subvention est un AWS service, le `GranteePrincipal` champ contient le principal du service, qui peut représenter plusieurs principaux bénéficiaires différents.

Par exemple, la commande suivante répertorie tous les octrois d'une clé KMS.

```
$  aws kms list-grants --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "Grants": [
        {
            "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1572216195.0,
            "GrantId":
 "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a",
            "Constraints": {
                "EncryptionContextSubset": {
                    "Department": "IT"
                }
            },
            "RetiringPrincipal": "arn:aws:iam::111122223333:role/adminRole",
```

```
            "Name": "",
            "IssuingAccount": "arn:aws:iam::111122223333:root",
            "GranteePrincipal": "arn:aws:iam::111122223333:user/exampleUser",
            "Operations": [
                "Decrypt"
            ]
        }
    ]
}
```

Utilisation d'un jeton d'octroi

L' AWS KMS API suit un modèle de cohérence éventuel. Lorsque vous créez un octroi, il se peut qu'il ne soit pas effectif immédiatement. Il se peut qu'il y ait un bref délai avant que le changement ne soit disponible via AWS KMS. La propagation de la modification dans l'ensemble du système prend généralement moins de quelques secondes, mais dans certains cas, cela peut prendre plusieurs minutes. Une fois que la modification a été entièrement appliquée à l'ensemble du système, le principal bénéficiaire peut utiliser les autorisations dans l'octroi sans spécifier le jeton d'octroi ou une preuve de l'octroi. Toutefois, si une subvention est si récente qu'elle n'est pas encore connue de tous AWS KMS, la demande peut échouer avec une `AccessDeniedException` erreur.

Pour utiliser immédiatement les autorisations dans un nouvel octroi, utilisez le jeton d'octroi pour l'octroi. Enregistrez le jeton de subvention renvoyé par l'CreateGrantopération. Soumettez ensuite le jeton de subvention dans la demande AWS KMS d'opération. Vous pouvez soumettre un jeton de subvention à n'importe quelle opération de AWS KMS subvention et vous pouvez soumettre plusieurs jetons de subvention dans la même demande.

L'exemple suivant utilise l'`CreateGrant`opération pour créer une autorisation autorisant les opérations GenerateDataKeyet Decrypt. Elle enregistre le jeton d'octroi que `CreateGrant` renvoie dans la variable `token`. Ensuite, dans un appel à l'opération `GenerateDataKey`, elle utilise le jeton d'octroi dans la variable `token`.

```
# Create a grant; save the grant token
$ token=$(aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:user/appUser \
    --retiring-principal arn:aws:iam::111122223333:user/acctAdmin \
    --operations GenerateDataKey Decrypt \
    --query GrantToken \
    --output text)
```

```
# Use the grant token in a request
$ aws kms generate-data-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-spec AES_256 \
    --grant-tokens $token
```

Les directeurs autorisés peuvent également utiliser un jeton de subvention pour retirer une nouvelle subvention avant même que la subvention ne soit disponible dans AWS KMS son intégralité. (L'opération `RevokeGrant` n'accepte pas de jeton d'octroi.) Pour plus de détails, consultez Retrait et révocation d'octrois.

```
# Retire the grant
$ aws kms retire-grant --grant-token $token
```

Retrait et révocation d'octrois

Pour supprimer un octroi, la retirer ou la révoquer.

Les RevokeGrantopérations RetireGrantet sont très similaires les unes aux autres. Les deux opérations suppriment un octroi, ce qui élimine les autorisations qu'il accorde. La principale différence entre ces opérations est la façon dont elles sont autorisées.

RevokeGrant

Comme pour la plupart AWS KMS des opérations, l'accès à l'`RevokeGrant`opération est contrôlé par le biais de politiques clés et de politiques IAM. L'RevokeGrantAPI peut être appelée par n'importe quel principal `kms:RevokeGrant` autorisé. Cette autorisation est incluse dans les autorisations standard accordées aux administrateurs de clé. En règle générale, les administrateurs révoquent un octroi pour refuser les autorisations qu'il accorde.

RetireGrant

L'octroi détermine qui peut la retirer. Cette conception vous permet de contrôler le cycle de vie d'un octroi sans modifier les politiques clé ou les politiques IAM. Généralement, vous retirez un octroi lorsque vous avez terminé d'utiliser ses autorisations.

Un octroi peut être retiré par un principal de retrait spécifié dans l'octroi. Le principal bénéficiaire peut également retirer l'octroi, mais seulement s'il est également un principal de retrait ou si l'octroi comprend l'opération `RetireGrant`. À titre de sauvegarde, le Compte AWS système dans lequel la subvention a été créée peut annuler la subvention.

Il existe une autorisation `kms:RetireGrant` qui peut être utilisée dans les politiques IAM, mais dont l'utilité est limitée. Les principaux spécifiés dans l'octroi peuvent retirer un octroi sans l'autorisation `kms:RetireGrant`. L'autorisation `kms:RetireGrant` à elle seule ne permet pas aux principaux de retirer un octroi. L'`kms:RetireGrant`autorisation n'est pas effective dans le cadre d'une politique clé ou d'une politique de contrôle des ressources.

- Pour refuser l'autorisation de retirer une subvention, vous pouvez utiliser une `Deny` action dont l'`kms:RetireGrant`autorisation figure dans vos politiques IAM.
- Le Compte AWS propriétaire de la clé KMS peut déléguer l'`kms:RetireGrant`autorisation à un principal IAM du compte.
- Si le mandant sortant est différent Compte AWS, les administrateurs de l'autre compte peuvent `kms:RetireGrant` déléguer l'autorisation de retirer la subvention à un directeur IAM de ce compte.

L' AWS KMS API suit un modèle de cohérence éventuel. Lorsque vous créez, retirez ou révoquez un octroi, il se peut qu'il y ait un bref délai avant que le changement ne soit disponible via AWS KMS. La propagation de la modification dans l'ensemble du système prend généralement moins de quelques secondes, mais dans certains cas, cela peut prendre plusieurs minutes. Si vous devez supprimer une nouvelle subvention immédiatement, avant qu'elle ne soit entièrement disponible AWS KMS, utilisez un jeton de subvention pour annuler la subvention. Vous ne pouvez pas utiliser un jeton d'octroi pour révoquer un octroi.

Clés de condition pour AWS KMS

Vous pouvez spécifier des conditions dans les politiques clés et les politiques IAM qui contrôlent l'accès aux AWS KMS ressources. L'Instruction de politique est en vigueur uniquement lorsque les conditions sont vérifiées. Par exemple, il est possible d'appliquer une instruction de politique après une date spécifique. Ou, vous pouvez faire en sorte qu'une instruction de politique contrôle l'accès uniquement lorsqu'une valeur spécifique apparaît dans une demande d'API.

Pour spécifier des conditions, utilisez les clés de condition dans l'élément `Condition` d'une instruction de politique avec des opérateurs de condition IAM. Certaines clés de condition s'appliquent de manière générale AWS ; d'autres sont spécifiques à AWS KMS.

Les valeurs des clés de condition doivent respecter les règles de caractères et de codage des politiques AWS KMS clés et des politiques IAM. Pour plus d'informations sur les règles de document de politique de clé, voir Format de politique de clé. Pour plus d'informations sur les règles du

document de politique IAM, veuillez consulter Exigences relatives aux noms IAM dans le Guide de l'utilisateur IAM.

Rubriques

- AWS clés de condition globales
- AWS KMS clés de condition
- AWS KMS clés de condition pour AWS Nitro Enclaves

AWS clés de condition globales

AWS définit des clés de condition globales, un ensemble de clés de conditions de politique pour tous les AWS services qui utilisent IAM pour le contrôle d'accès. AWS KMS prend en charge toutes les clés de condition globales. Vous pouvez les utiliser dans les politiques AWS KMS clés et les politiques IAM.

Par exemple, vous pouvez utiliser la clé de condition PrincipalArn globale aws : pour autoriser l'accès à une AWS KMS key (clé KMS) uniquement lorsque le principal de la demande est représenté par le nom de ressource Amazon (ARN) dans la valeur de la clé de condition. Pour prendre en charge le contrôle d'accès basé sur les attributs (ABAC) dans AWS KMS, vous pouvez utiliser la clé de condition globale aws :ResourceTag/tag-key dans une politique IAM afin d'autoriser l'accès aux clés KMS avec une balise particulière.

Pour éviter qu'un AWS service ne soit utilisé comme un sous-traitant confus dans une politique où le principal est un directeur de AWS service, vous pouvez utiliser les clés de condition aws:SourceArnou aws:SourceAccountglobales. Pour en savoir plus, consultez Utilisation des clés de condition `aws:SourceArn` ou `aws:SourceAccount`.

Pour plus d'informations sur les clés de condition AWS globales, y compris les types de demandes dans lesquels elles sont disponibles, voir Clés contextuelles de conditions AWS globales dans le guide de l'utilisateur IAM. Pour accéder à des exemples d'utilisation des clés de condition globale dans des politiques IAM, veuillez consulter Contrôle de l'accès aux demandes et Contrôle des clés de balise dans le Guide de l'utilisateur IAM.

Les rubriques suivantes fournissent des conseils spéciaux pour l'utilisation des clés de condition basées sur les adresses IP et les points de terminaison VPC.

Rubriques

- Utilisation de la condition d'adresse IP dans les politiques avec autorisations AWS KMS
- Utilisation de conditions de point de terminaison d'un VPC dans des politiques avec des autorisations AWS KMS
- Utilisation IPv6 des adresses dans les politiques clés IAM et KMS

Utilisation de la condition d'adresse IP dans les politiques avec autorisations AWS KMS

Vous pouvez l'utiliser AWS KMS pour protéger vos données dans le cadre d'un AWS service intégré. Mais soyez prudent lorsque vous spécifiez les opérateurs de condition d'adresse IP ou la clé de `aws:SourceIp` condition dans la même déclaration de politique qui autorise ou refuse l'accès à AWS KMS. Par exemple, la politique décrite dans AWS: Refuse l'accès à la AWS base de l'adresse IP source limite les AWS actions aux demandes provenant de la plage d'adresses IP spécifiée.

Envisagez le scénario suivant :

1. Vous associez une politique telle que celle présentée à l'adresse AWS suivante : Refuse l'accès à une identité IAM en AWS fonction de l'adresse IP source. Vous définissez la valeur de la clé de condition `aws:SourceIp` sur la plage d'adresses IP de la société de l'utilisateur. Cette identité IAM est associée à d'autres politiques qui lui permettent d'utiliser Amazon EBS EC2, Amazon et. AWS KMS
2. L'identité tente d'attacher un volume EBS chiffré à une EC2 instance. Cette action échoue avec une erreur d'autorisation bien que l'utilisateur soit autorisé à utiliser l'ensemble des services concernés.

L'étape 2 échoue car la demande AWS KMS de déchiffrement de la clé de données cryptée du volume provient d'une adresse IP associée à l' EC2 infrastructure Amazon. Pour que l'opération aboutisse, la requête doit provenir de l'adresse IP de l'utilisateur d'origine. Étant donné que la politique de l'étape 1 refuse explicitement toutes les demandes provenant d'adresses IP autres que celles spécifiées, Amazon EC2 n'est pas autorisé à déchiffrer la clé de données cryptée du volume EBS.

Par ailleurs, la clé de condition `aws:SourceIP` n'est pas en vigueur lorsque la demande provient d'un point de terminaison d'un VPC Amazon. Pour restreindre les demandes à un point de terminaison VPC, y compris à un point de terminaison VPC AWS KMS, utilisez les clés de condition `aws:SourceVpce` ou `aws:SourceVpc`. Pour plus d'informations, consultez Points de terminaison

d'un VPC - Contrôle de l'utilisation de points de terminaison dans le manuel Guide d'utilisateur Amazon VPC.

Utilisation de conditions de point de terminaison d'un VPC dans des politiques avec des autorisations AWS KMS

AWS KMS prend en charge les points de terminaison Amazon Virtual Private Cloud (Amazon VPC) alimentés par. AWS PrivateLink Vous pouvez utiliser les clés de condition globales suivantes dans les politiques clés et les politiques IAM pour contrôler l'accès aux AWS KMS ressources lorsque la demande provient d'un VPC ou utilise un point de terminaison VPC. Pour plus de détails, veuillez consulter Utiliser les points de terminaison VPC pour contrôler l'accès aux ressources AWS KMS.

- `aws:SourceVpc` limite l'accès aux requêtes à partir du VPC spécifié.
- `aws:SourceVpce` limite l'accès aux requêtes à partir du point de terminaison d'un VPC spécifié.

Si vous utilisez ces clés de condition pour contrôler l'accès aux clés KMS, vous risquez de refuser par inadvertance l'accès aux AWS services utilisés en votre AWS KMS nom.

Prenez soin d'éviter une situation comme dans l'exemple des clés de condition d'adresse IP. Si vous limitez les demandes de clé KMS à un point de terminaison VPC ou VPC, les appels AWS KMS provenant d'un service intégré, tel qu'Amazon S3 ou Amazon EBS, risquent d'échouer. Cela peut se produire même si la requête source provient au final du VPC ou du point de terminaison d'un VPC.

Utilisation IPv6 des adresses dans les politiques clés IAM et KMS

Avant d'essayer d'accéder à KMS via KMS IPv6, assurez-vous que toutes les politiques de clé et d'IAM contenant des restrictions d'adresse IP sont mises à jour pour inclure les plages d' IPv6 adresses. Les politiques basées sur les adresses IP qui ne sont pas mises à jour pour gérer IPv6 les adresses peuvent entraîner une perte ou un accès incorrect des clients lorsqu'ils commencent à les utiliser IPv6. Pour obtenir des conseils généraux sur les contrôles d'accès KMS, voirAccès aux clés KMS et autorisations. Pour en savoir plus sur la prise en charge du KMS et du double stack, consultezSupport des terminaux à double pile.

> **Important**
>
> Ces déclarations n'autorisent aucune action. Utilisez ces instructions en combinaison avec d'autres instructions qui autorisent des actions spécifiques.

La déclaration suivante refuse explicitement l'accès à toutes les autorisations KMS pour les demandes provenant de la `192.0.2.*` plage d' IPv4 adresses. Les autorisations KMS ne sont pas explicitement refusées aux adresses IP situées en dehors de cette plage. Comme toutes les IPv6 adresses se situent en dehors de la plage refusée, cette instruction ne refuse pas explicitement les autorisations KMS pour aucune IPv6 adresse.

```
{
     "Sid": "DenyKMSPermissions",
     "Effect": "Deny",
    "Action": [
        "kms:*"
    ],
    "Resource": "*",
    "Condition": {
        "NotIpAddress": {
            "aws:SourceIp": [
                "192.0.2.0/24"
            ]
        }
    }
}
```

Vous pouvez modifier l'`Condition`élément pour refuser à la fois les plages d'adresses IPv4 IPv6 (`192.0.2.0/24`2001:db8:1234::/32) et (), comme indiqué dans l'exemple suivant.

```
{
    "Sid": "DenyKMSPermissions",
    "Effect": "Deny",
    "Action": [
        "kms:*"
    ],
    "Resource": "*",
    "Condition": {
        "NotIpAddress": {
            "aws:SourceIp": [
                "192.0.2.0/24",
                "2001:db8:1234::/32"
            ]
        }
    }
}
```

AWS KMS clés de condition

AWS KMS fournit un ensemble de clés de condition que vous pouvez utiliser dans les politiques clés et les politiques IAM. Ces clés de condition sont spécifiques à AWS KMS. Par exemple, vous pouvez utiliser la clé de condition `kms:EncryptionContext:`*`context-key`* pour exiger un contexte de chiffrement particulier lorsque vous contrôlez l'accès à une clé KMS de chiffrement symétrique.

Conditions pour une demande d'opération d'API

De nombreuses clés de AWS KMS condition contrôlent l'accès à une clé KMS en fonction de la valeur d'un paramètre dans la demande d' AWS KMS opération. Par exemple, vous pouvez utiliser la clé de KeySpec condition kms : dans une politique IAM pour autoriser l'utilisation de l'CreateKeyopération uniquement lorsque la valeur du `KeySpec` paramètre de la `CreateKey` demande est`RSA_4096`.

Ce type de condition fonctionne même lorsque le paramètre n'apparaît pas dans la demande, par exemple lorsque vous utilisez la valeur par défaut du paramètre. Par exemple, vous pouvez utiliser la clé de KeySpec condition kms : pour permettre aux utilisateurs d'utiliser l'`CreateKey`opération uniquement lorsque la valeur du `KeySpec` paramètre est`SYMMETRIC_DEFAULT`, qui est la valeur par défaut. Cette condition autorise les demandes dont le paramètre `KeySpec` a pour valeur `SYMMETRIC_DEFAULT` et les demandes qui n'ont pas de paramètre `KeySpec`.

Conditions pour les clés KMS utilisées dans les opérations d'API

Certaines clés de AWS KMS condition peuvent contrôler l'accès aux opérations en fonction d'une propriété de la clé KMS utilisée dans l'opération. Par exemple, vous pouvez utiliser la KeyOrigin condition kms : pour autoriser les principaux à appeler GenerateDataKeyune clé KMS uniquement lorsque `Origin` la clé KMS est`AWS_KMS`. Pour savoir si une clé de condition peut être utilisée de cette manière, consultez la description de la clé de condition.

L'opération doit être une opération de ressource de clé KMS, c'est-à-dire une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS  key` dans la colonne `Resources` de l'opération. Si vous utilisez ce type de clé de condition avec une opération qui n'est pas autorisée pour une ressource clé KMS particulière, par exemple ListKeys, l'autorisation n'est pas effective car la condition ne peut jamais être satisfaite. Aucune ressource de clé KMS n'est impliquée dans l'autorisation de l'opération `ListKeys` ni aucune propriété `KeySpec`.

Les rubriques suivantes décrivent chaque clé de AWS KMS condition et incluent des exemples de déclarations de politique illustrant la syntaxe des politiques.

Utilisation d'opérateurs d'ensemble avec des clés de condition

Lorsqu'une condition de stratégie compare deux ensembles de valeurs, tels que le jeu de balises d'une demande et le jeu de balises d'une politique, vous devez indiquer AWS comment comparer les ensembles. IAM définit deux opérateurs d'ensemble, `ForAnyValue` et `ForAllValues`, à cette fin. Utilisez les opérateurs d'ensemble uniquement avec des clés de condition multi-valeurs, qui les nécessitent. N'utilisez pas d'opérateurs d'ensemble avec des clés de condition à valeur unique. Comme toujours, testez vos instructions de politiques de manière approfondie avant de les utiliser au sein d'un environnement de production.

Les clés de condition sont à valeur unique ou multi-valeurs. Pour déterminer si une clé de AWS KMS condition est à valeur unique ou à valeurs multiples, consultez la colonne Type de valeur dans la description de la clé de condition.

- Les clés de condition à valeur unique ont au plus une valeur dans le contexte d'autorisation (la demande ou la ressource). Par exemple, étant donné que chaque appel d'API ne peut provenir que d'une seule Compte AWS, kms : CallerAccount est une clé de condition à valeur unique. N'utilisez pas d'opérateur d'ensemble avec une clé de condition à valeur unique.
- Les clés de condition multi-valeurs ont plusieurs valeurs dans le contexte d'autorisation (la demande ou la ressource). Par exemple, étant donné que chaque clé KMS peut avoir plusieurs alias, kms : ResourceAliases peut avoir plusieurs valeurs. Les clés de condition multi-valeurs nécessitent un opérateur d'ensemble.

Notez que la différence entre les clés de condition à valeur unique et multi-valeurs dépend du nombre de valeurs dans le contexte d'autorisation, et non du nombre de valeurs dans la condition de politique.

> **Warning**
>
> L'utilisation d'un opérateur d'ensemble avec une clé de condition à valeur unique peut créer une instruction de politique trop permissive (ou trop restrictive). Utilisez des opérateurs d'ensemble uniquement avec des clés de condition multi-valeurs.
> Si vous créez ou mettez à jour une politique qui inclut un opérateur `ForAllValues` défini avec les clés de contexte ou de `aws:RequestTag/`*`tag-key`* condition, AWS KMS renvoie le message d'erreur suivant kms: EncryptionContext :
> `OverlyPermissiveCondition: Using the ForAllValues set operator with a single-valued condition key matches requests without the specified`

```
    [encryption context or tag] or with an unspecified [encryption
    context or tag]. To fix, remove ForAllValues.
```

Pour de plus amples informations sur les opérateurs d'ensemble `ForAnyValue` et `ForAllValues`, veuillez consulter Utilisation de plusieurs clés et valeurs dans le Guide de l'utilisateur IAM. Pour plus d'informations sur le risque lié à l'utilisation de l'opérateur `ForAllValues` défini avec une condition à valeur unique, voir Avertissement de sécurité : ForAllValues clé à valeur unique dans le guide de l'utilisateur IAM.

Rubriques

- km : BypassPolicyLockoutSafetyCheck
- km : CallerAccount
- kms : CustomerMasterKeySpec (obsolète)
- kms : CustomerMasterKeyUsage (obsolète)
- km : DataKeyPairSpec
- km : EncryptionAlgorithm
- kms EncryptionContext : touche contextuelle
- km : EncryptionContextKeys
- km : ExpirationModel
- km : GrantConstraintType
- km : GrantIsFor AWSResource
- km : GrantOperations
- km : GranteePrincipal
- km : KeyAgreementAlgorithm
- km : KeyOrigin
- km : KeySpec
- km : KeyUsage
- km : MacAlgorithm
- km : MessageType
- km : MultiRegion

- km : MultiRegionKeyType
- km : PrimaryRegion
- km : ReEncryptOnSameKey
- km : RequestAlias
- km : ResourceAliases
- km : ReplicaRegion
- km : RetiringPrincipal
- km : RotationPeriodInDays
- km : ScheduleKeyDeletionPendingWindowInDays
- km : SigningAlgorithm
- km : ValidTo
- km : ViaService
- km : WrappingAlgorithm
- km : WrappingKeySpec

km : BypassPolicyLockoutSafetyCheck

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:BypassPolicyLockoutSafetyCheck`	Booléen	À valeur unique	`CreateKey` `PutKeyPolicy`	Politiques IAM uniquement Politiques de clé et politiques IAM

La clé de `kms:BypassPolicyLockoutSafetyCheck` condition contrôle l'accès aux PutKeyPolicyopérations CreateKeyet en fonction de la valeur du `BypassPolicyLockoutSafetyCheck` paramètre dans la demande.

L'exemple d'instruction de politique IAM suivant empêche les utilisateurs de contourner le contrôle de sécurité du verrouillage de la politique en leur refusant l'autorisation de créer des clés KMS lorsque

la valeur du paramètre `BypassPolicyLockoutSafetyCheck` dans la demande `CreateKey` est `true`..

```
{
  "Effect": "Deny",
  "Action": [
    "kms:CreateKey",
    "kms:PutKeyPolicy"
  ],
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:BypassPolicyLockoutSafetyCheck": true
    }
  }
}
```

Vous pouvez également utiliser la clé de condition `kms:BypassPolicyLockoutSafetyCheck` dans une politique IAM ou une politique de clé pour contrôler l'accès à l'opération `PutKeyPolicy`. L'exemple d'instruction de politique suivant dans une politique de clé empêche les utilisateurs de contourner le contrôle de sécurité du verrouillage de la politique lors de la modification de la politique d'une clé KMS.

Plutôt que d'utiliser explicitement `Deny`, cette instruction de politique utilise `Allow` avec l'opérateur de condition Null afin d'autoriser l'accès uniquement lorsque la demande n'inclut pas le paramètre `BypassPolicyLockoutSafetyCheck`. Lorsque le paramètre n'est pas utilisé, la valeur par défaut est `false`. Cette instruction de politique légèrement affaiblie peut être remplacée dans le cas rare où un contournement est nécessaire.

```
{
  "Effect": "Allow",
  "Action": "kms:PutKeyPolicy",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:BypassPolicyLockoutSafetyCheck": true
    }
  }
}
```

Voir aussi

- km : KeySpec
- km : KeyOrigin
- km : KeyUsage

km : CallerAccount

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:CallerAccount`	Chaîne	À valeur unique	Opérations liées aux ressources de clé KMS Opérations liées au magasin de clés personnalisé	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour autoriser ou refuser l'accès à toutes les identités (utilisateurs et rôles) d'un Compte AWS. Dans les politiques de clé, vous utilisez l'élément `Principal` pour spécifier les identités auxquelles l'instruction de politique s'applique. La syntaxe de l'élément `Principal` ne permet pas de spécifier toutes les identités dans un Compte AWS. Mais vous pouvez obtenir cet effet en combinant cette clé de condition avec un `Principal` élément qui spécifie toutes les AWS identités.

Vous pouvez l'utiliser pour contrôler l'accès à toute opération de ressource clé KMS, c'est-à-dire toute AWS KMS opération utilisant une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS key` dans la colonne `Resources` de l'opération. Elle est également valable pour les opérations qui gèrent des magasins de clés personnalisés.

Par exemple, l'instruction de politique suivante montre comment utiliser la clé de condition `kms:CallerAccount`. Cette déclaration de politique fait partie de la politique clé Clé gérée par AWS d'Amazon EBS. Il combine un `Principal` élément qui spécifie toutes les AWS identités avec la clé de `kms:CallerAccount` condition pour autoriser efficacement l'accès à toutes les identités dans Compte AWS 111122223333. Il contient une clé de AWS KMS condition supplémentaire

(`kms:ViaService`) pour limiter davantage les autorisations en n'autorisant que les demandes provenant d'Amazon EBS. Pour de plus amples informations, veuillez consulter km : ViaService.

```
{
  "Sid": "Allow access through EBS for all principals in the account that are
 authorized to use EBS",
  "Effect": "Allow",
  "Principal": {"AWS": "*"},
  "Condition": {
    "StringEquals": {
      "kms:CallerAccount": "111122223333",
      "kms:ViaService": "ec2.us-west-2.amazonaws.com"
    }
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:CreateGrant",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

kms : CustomerMasterKeySpec (obsolète)

La clé de condition `kms:CustomerMasterKeySpec` est obsolète. Utilisez plutôt la clé de KeySpec condition kms :.

Les clés de condition `kms:CustomerMasterKeySpec` et `kms:KeySpec` fonctionnent de la même manière. Seuls les noms diffèrent. Nous vous recommandons d'utiliser `kms:KeySpec`. Toutefois, pour éviter d'interrompre les modifications, AWS KMS prend en charge les deux clés de condition.

kms : CustomerMasterKeyUsage (obsolète)

La clé de condition `kms:CustomerMasterKeyUsage` est obsolète. Utilisez plutôt la clé de KeyUsage condition kms :.

Les clés de condition `kms:CustomerMasterKeyUsage` et `kms:KeyUsage` fonctionnent de la même manière. Seuls les noms diffèrent. Nous vous recommandons d'utiliser `kms:KeyUsage`.

Toutefois, pour éviter d'interrompre les modifications, AWS KMS prend en charge les deux clés de condition.

km : DataKeyPairSpec

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:DataKeyPairSpec`	Chaîne	À valeur unique	`GenerateDataKeyPair` `GenerateDataKeyPairWithoutPlaintext`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour contrôler l'accès aux GenerateDataKeyPairWithoutPlaintextopérations GenerateDataKeyPairet en fonction de la valeur du `KeyPairSpec` paramètre dans la demande. Par exemple, vous pouvez autoriser des utilisateurs à générer uniquement des types particuliers de paires de clés de données.

L'exemple suivant d'instruction de politique de clé utilise la clé de condition `kms:DataKeyPairSpec` pour autoriser les utilisateurs à utiliser la clé KMS afin de générer uniquement des paires de clés de données RSA.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:GenerateDataKeyPair",
    "kms:GenerateDataKeyPairWithoutPlaintext"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:DataKeyPairSpec": "RSA*"
    }
```

```
  }
}
```

Voir aussi

- km : KeySpec
- the section called “km : EncryptionAlgorithm”
- the section called “kms EncryptionContext : touche contextuelle”
- the section called “km : EncryptionContextKeys”

km : EncryptionAlgorithm

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d’API	Type de stratégie

`kms:EncryptionAlgorithm`	Chaîne	À valeur unique	`Decrypt`
			`Encrypt`
			`GenerateDataKey`
			`GenerateDataKeyPair`
			`GenerateDataKeyPairWithoutPlaintext`
			`GenerateDataKeyWithoutPlaintext`
			`ReEncrypt`
				Politiques de clé et politiques IAM

Vous pouvez utiliser la clé de condition `kms:EncryptionAlgorithm` pour contrôler l'accès aux opérations de chiffrement en fonction de l'algorithme de chiffrement utilisé dans l'opération. Pour les ReEncryptopérations de chiffrement, de déchiffrement et de déchiffrement, il contrôle l'accès en fonction de la valeur du EncryptionAlgorithmparamètre figurant dans la demande. Pour les opérations qui génèrent des clés de données et des paires de clés de données, elle contrôle l'accès en fonction de l'algorithme de chiffrement utilisé pour chiffrer la clé de données.

Cette clé de condition n'a aucun effet sur les opérations effectuées en dehors de AWS KMS, telles que le chiffrement avec la clé publique dans une paire de clés KMS asymétrique en dehors de. AWS KMS

EncryptionAlgorithm paramètre dans une demande

Pour autoriser les utilisateurs à utiliser uniquement un algorithme de chiffrement particulier avec une clé KMS, utilisez une instruction de politique avec un effet `Deny` et un opérateur de condition `StringNotEquals`. Par exemple, l'exemple suivant d'instruction de politique de clé interdit aux principaux pouvant endosser le rôle `ExampleRole` d'utiliser cette clé KMS dans les opérations de chiffrement spécifiées, sauf si l'algorithme de chiffrement de la demande est `RSAES_OAEP_SHA_256`. un algorithme de chiffrement asymétrique utilisé avec des clés KMS RSA.

Contrairement à une instruction de politique permettant à un utilisateur d'utiliser un algorithme de chiffrement particulier, une instruction de politique avec un double négatif comme celui-ci empêche les autres politiques et octrois associés à cette clé KMS d'autoriser ce rôle à utiliser d'autres algorithmes de chiffrement. Le paramètre `Deny` dans cette instruction de politique de clé prévaut sur toute politique de clé ou politique IAM ayant un effet `Allow`. Il prévaut également sur tous les octrois associés à cette clé KMS et à ses principaux.

```
{
  "Sid": "Allow only one encryption algorithm with this asymmetric KMS key",
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*"
  ],
  "Resource": "*",
  "Condition": {
```

```
    "StringNotEquals": {
      "kms:EncryptionAlgorithm": "RSAES_OAEP_SHA_256"
    }
  }
}
```

Algorithme de chiffrement utilisé pour l'opération

Vous pouvez également utiliser la clé de condition `kms:EncryptionAlgorithm` pour contrôler l'accès aux opérations en fonction de l'algorithme de chiffrement utilisé dans l'opération, même lorsque l'algorithme n'est pas spécifié dans la demande. Cela vous permet d'exiger ou d'interdire l'algorithme `SYMMETRIC_DEFAULT`, qui peut ne pas être spécifié dans une demande, car il s'agit de la valeur par défaut.

Cette fonction vous permet également d'utiliser la clé de condition `kms:EncryptionAlgorithm` pour contrôler l'accès aux opérations qui génèrent des clés de données et des paires de clés de données. Ces opérations utilisent uniquement des clés KMS de chiffrement symétriques et l'algorithme `SYMMETRIC_DEFAULT`.

Par exemple, cette politique IAM limite ses principaux au chiffrement symétrique. Elle interdit l'accès à toute clé KMS dans l'exemple de compte pour les opérations de chiffrement, sauf si l'algorithme de chiffrement spécifié dans la demande ou utilisé dans l'opération est SYMMETRIC_DEFAULT. Y compris GenerateDataKeyles `GenerateDataKey*` ajouts GenerateDataKeyWithoutPlaintextGenerateDataKeyPair,, et GenerateDataKeyPairWithoutPlaintextaux autorisations. La condition n'a aucun effet sur ces opérations, car elles utilisent toujours un algorithme de chiffrement symétrique.

```
{
  "Sid": "AllowOnlySymmetricAlgorithm",
  "Effect": "Deny",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringNotEquals": {
      "kms:EncryptionAlgorithm": "SYMMETRIC_DEFAULT"
    }
```

```
  }
}
```

Voir aussi

- the section called “km : MacAlgorithm”
- km : SigningAlgorithm

kms EncryptionContext : touche contextuelle

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d’API	Type de stratégie

`kms:EncryptionContext: context-key`	Chaîne	À valeur unique	`CreateGrant`
			`Encrypt`
			`Decrypt`
			`GenerateDataKey`
			`GenerateDataKeyPair`
			`GenerateDataKeyPairWithoutPlaintext`
			`GenerateDataKeyWithoutPlaintext`
			`ReEncrypt`
			`RetireGrant`
				Politiques de clé et politiques IAM

Vous pouvez utiliser la clé de condition `kms:EncryptionContext:context-key` pour contrôler l'accès à une clé KMS de chiffrement symétrique en fonction du contexte de chiffrement d'une demande d'opération de chiffrement. Utilisez cette clé de condition pour évaluer à la fois la clé et la valeur dans la paire de contexte de chiffrement. Pour évaluer uniquement les clés de contexte de chiffrement ou pour exiger un contexte de chiffrement indépendamment des clés ou des valeurs, utilisez la clé de EncryptionContextKeys condition kms :.

> **Note**
>
> Les valeurs clé de condition doivent respecter les règles de caractère pour les politiques de clé et les politiques IAM. Certains caractères valides dans un contexte de chiffrement ne sont pas valides dans les politiques. Vous ne pourrez peut-être pas utiliser cette clé de condition pour exprimer toutes les valeurs du contexte de chiffrement valides. Pour plus d'informations sur les règles de document de politique de clé, voir Format de politique de clé. Pour plus d'informations sur les règles du document de politique IAM, veuillez consulter Exigences relatives aux noms IAM dans le Guide de l'utilisateur IAM.

Vous ne pouvez pas spécifier de contexte de chiffrement dans une opération de chiffrement avec une clé KMS asymétrique ou une clé KMS HMAC. Les algorithmes asymétriques et les algorithmes MAC ne prennent pas en charge un contexte de chiffrement.

Pour utiliser la clé de condition kms: EncryptionContext : clé de contexte, remplacez l'*context-key*espace réservé par la clé de contexte de chiffrement. Remplacez l'espace réservé *context-value* par la valeur du contexte de chiffrement.

```
"kms:EncryptionContext:context-key": "context-value"
```

Par exemple, la clé de condition suivante spécifie un contexte de chiffrement dans lequel la clé est `AppName` et la valeur est `ExampleApp` (`AppName = ExampleApp`).

```
"kms:EncryptionContext:AppName": "ExampleApp"
```

Il s'agit d'une clé de condition à valeur unique. La clé de la clé de condition spécifie une clé de contexte de chiffrement particulière (context-key). Bien que vous puissiez inclure plusieurs paires de contexte de chiffrement dans chaque demande d'API, la paire de contexte de chiffrement avec le paramètre context-key ne peut avoir qu'une seule valeur. Par exemple, la clé de condition `kms:EncryptionContext:Department` s'applique uniquement aux paires de contexte de

chiffrement avec une clé `Department`, et toute paire de contexte de chiffrement donnée avec la clé `Department` ne peut avoir qu'une seule valeur.

N'utilisez pas d'opérateur d'ensemble avec la clé de condition `kms:EncryptionContext:`*`context-key`*. Si vous créez une instruction de politique avec une action `Allow`, la clé de condition `kms:EncryptionContext:`*`context-key`* et l'opérateur d'ensemble `ForAllValues`, la condition autorise les demandes sans contexte de chiffrement et les demandes avec des paires de contexte de chiffrement qui ne sont pas spécifiées dans la condition de politique.

> **Warning**
>
> N'utilisez pas d'opérateur d'ensemble `ForAnyValue` ou `ForAllValues` avec cette clé de condition à valeur unique. Ces opérateurs d'ensemble peuvent créer une condition de politique qui ne nécessite pas de valeurs que vous avez l'intention d'exiger et autorise les valeurs que vous avez l'intention d'interdire.
> Si vous créez ou mettez à jour une politique qui inclut un opérateur `ForAllValues` set avec la touche de contexte kms: EncryptionContext :, AWS KMS renvoie le message d'erreur suivant :
> `OverlyPermissiveCondition:EncryptionContext: Using the ForAllValues set operator with a single-valued condition key matches requests without the specified encryption context or with an unspecified encryption context. To fix, remove ForAllValues.`

Pour exiger une paire de contexte de chiffrement particulière, utilisez la clé de condition `kms:EncryptionContext:`*`context-key`* avec l'opérateur `StringEquals`.

L'exemple d'instruction de politique de clé suivant autorise les principaux qui peuvent endosser le rôle à utiliser la clé KMS dans une demande `GenerateDataKey`, uniquement lorsque le contexte de chiffrement de la demande inclut la paire `AppName:ExampleApp`. D'autres paires de contexte de chiffrement sont autorisées.

Le nom de la clé n'est pas sensible à la casse. La sensibilité à la casse de la valeur est déterminée par l'opérateur de condition, tel que `StringEquals`. Pour plus de détails, veuillez consulter Sensibilité à la casse de la condition de contexte de chiffrement.

```
{
```

```
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:AppName": "ExampleApp"
    }
  }
}
```

Pour exiger une paire de contextes de chiffrement et interdire toutes les autres paires de contextes de chiffrement, utilisez à la fois kms: EncryptionContext : clé de contexte et kms:EncryptionContextKeysdans la déclaration de politique. L'exemple d'instruction de politique suivant utilise la condition `kms:EncryptionContext:AppName` pour exiger la présence de la paire de contexte de chiffrement `AppName=ExampleApp` dans la demande. Il utilise également une clé de condition `kms:EncryptionContextKeys` avec l'opérateur d'ensemble `ForAllValues` pour autoriser uniquement la clé de contexte de chiffrement `AppName`.

L'opérateur d'ensemble `ForAllValues` limite les clés de contexte de chiffrement dans la demande à `AppName`. Si la condition `kms:EncryptionContextKeys` avec l'opérateur d'ensemble `ForAllValues` a été utilisée seule dans une instruction de politique, cet opérateur d'ensemble autoriserait les demandes sans contexte de chiffrement. Toutefois, si la demande n'avait pas de contexte de chiffrement, la condition `kms:EncryptionContext:AppName` échouerait. Pour plus de détails sur l'opérateur d'ensemble `ForAllValues`, veuillez consulter Utilisation de plusieurs clés et valeurs dans le Guide de l'utilisateur IAM.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/KeyUsers"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:AppName": "ExampleApp"
    },
    "ForAllValues:StringEquals": {
```

```
      "kms:EncryptionContextKeys": [
        "AppName"
      ]
    }
  }
}
```

Vous pouvez également utiliser cette clé de condition pour refuser l'accès à une clé KMS pour une opération particulière. L'exemple d'instruction de politique de clé suivant utilise un effet `Deny` pour interdire au principal d'utiliser la clé KMS si le contexte de chiffrement de la demande inclut une paire de contexte de chiffrement `Stage=Restricted`. Cette condition permet une demande avec d'autres paires de contexte de chiffrement, y compris les paires de contexte de chiffrement avec la clé `Stage` et d'autres valeurs, telles que `Stage=Test`.

```
{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Stage": "Restricted"
    }
  }
}
```

Utilisation de plusieurs paires de contexte de chiffrement

Vous pouvez exiger ou interdire plusieurs paires de contexte de chiffrement. Vous pouvez également exiger l'une des différentes paires de contexte de chiffrement. Pour plus de détails sur la logique utilisée pour interpréter ces conditions, veuillez consulter Création d'une condition avec plusieurs clés ou valeurs dans le Guide de l'utilisateur IAM.

> **Note**
>
> Les versions antérieures de cette rubrique affichaient des déclarations de politique qui utilisaient les opérateurs `ForAnyValue` et `ForAllValues` set avec la clé de condition kms: EncryptionContext : touche contextuelle. L'utilisation d'un opérateur d'ensemble avec une clé

de condition à valeur unique peut entraîner des politiques qui autorisent des demandes sans contexte de chiffrement et des paires de contexte de chiffrement non spécifiées.
Par exemple, une condition de politique avec l'effet `Allow`, l'opérateur d'ensemble `ForAllValues` et la clé de condition `"kms:EncryptionContext:Department": "IT"` ne limite pas le contexte de chiffrement à la paire « Department = IT ». Elle autorise les demandes sans contexte de chiffrement et les demandes avec des paires de contexte de chiffrement non spécifiées, telles que `Stage=Restricted`.
Veuillez revoir vos politiques et éliminer l'opérateur défini de toute condition avec kms: EncryptionContext : touche contextuelle. Les tentatives de création ou de mise à jour d'une politique avec ce format échouent avec une exception `OverlyPermissiveCondition`. Pour résoudre l'erreur, supprimez l'opérateur d'ensemble.

Pour exiger plusieurs paires de contexte de chiffrement, répertoriez les paires dans la même condition. L'exemple d'instruction de politique de clé suivant nécessite deux paires de contexte de chiffrement, `Department=IT` et `Project=Alpha`. Puisque les conditions ont des clés différentes (`kms:EncryptionContext:Department` et `kms:EncryptionContext:Project`), elles sont implicitement connectées par un opérateur AND. D'autres paires de contexte de chiffrement sont autorisées, mais non exigées.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Department": "IT",
      "kms:EncryptionContext:Project": "Alpha"
    }
  }
}
```

Pour exiger une paire de contexte de chiffrement OU une autre paire, placez chaque clé de condition dans une instruction de politique distincte. L'exemple de politique de clé suivant nécessite des paires `Department=IT` ou `Project=Alpha`, ou les deux. D'autres paires de contexte de chiffrement sont autorisées, mais non exigées.

```
{
 "Effect": "Allow",
 "Principal": {
  "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
 },
 "Action": "kms:GenerateDataKey",
 "Resource": "*",
 "Condition": {
  "StringEquals": {
   "kms:EncryptionContext:Department": "IT"
  }
 }
},
{
 "Effect": "Allow",
 "Principal": {
  "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
 },
 "Action": "kms:GenerateDataKey",
 "Resource": "*",
 "Condition": {
  "StringEquals": {
   "kms:EncryptionContext:Project": "Alpha"
  }
 }
}
```

Pour exiger des paires de chiffrement particulières et exclure toutes les autres paires de contextes de chiffrement, utilisez à la fois kms: EncryptionContext : clé de contexte et `kms:EncryptionContextKeys`dans la déclaration de politique. La déclaration de politique clé suivante utilise la condition kms: EncryptionContext :context-key pour exiger un contexte de chiffrement avec `Department=IT` les `Project=Alpha` deux paires. Elle utilise une clé de condition `kms:EncryptionContextKeys` avec l'opérateur d'ensemble `ForAllValues` pour n'autoriser que les clés de contexte de chiffrement `Department` et `Project`.

L'opérateur d'ensemble `ForAllValues` limite les clés de contexte de chiffrement dans la demande à `Department` et `Project`. S'il était utilisé seul dans une condition, cet opérateur set autoriserait les requêtes sans contexte de chiffrement, mais dans cette configuration, la clé de contexte kms: EncryptionContext : échouerait dans cette condition.

```
{
```

```
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Department": "IT",
      "kms:EncryptionContext:Project": "Alpha"
    },
    "ForAllValues:StringEquals": {
      "kms:EncryptionContextKeys": [
        "Department",
        "Project"
      ]
    }
  }
}
```

Vous pouvez également interdire plusieurs paires de contexte de chiffrement. L'exemple d'instruction de politique de clé suivant utilise un effet `Deny` pour interdire au principal d'utiliser les clés KMS si le contexte de chiffrement de la demande inclut une paire `Stage=Restricted` ou `Stage=Production`.

Plusieurs valeurs (`Restricted` et `Production`) pour la même clé (`kms:EncryptionContext:Stage`) sont implicitement connectés par un OR. Pour plus de détails, veuillez consulter Logique d'évaluation pour les conditions avec plusieurs clés ou valeurs dans le Guide de l'utilisateur IAM.

```
{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Stage": [
         "Restricted",
         "Production"
```

```
      ]
    }
  }
}
```

Sensibilité à la casse de la condition de contexte de chiffrement

Le contexte de chiffrement indiqué dans une opération de déchiffrement doit correspondre exactement au contexte de chiffrement précisé dans l'opération de chiffrement (en tenant compte des minuscules/majuscules). Seul l'ordre des paires dans un contexte de chiffrement avec plusieurs paire peut varier.

En revanche, dans les conditions des politiques, la clé de condition n'est pas sensible à la casse. Le respect de la casse de la valeur de condition est déterminée par l'opérateur de condition de politique que vous utilisez, comme `StringEquals` ou `StringEqualsIgnoreCase`.

À ce titre, la clé de condition, qui comprend le préfixe `kms:EncryptionContext:` et le remplacement *`context-key`*, n'est pas sensible à la casse. Une politique qui utilise cette condition ne vérifie pas la casse des éléments de la clé de condition. Le respect de la casse de la valeur, à savoir, le remplacement *`context-value`*, est déterminé par l'opérateur de condition de la politique.

Par exemple, l'instruction de politique suivante autorise l'opération lorsque le contexte de chiffrement inclut une clé `Appname`, quelle que soit sa capitalisation. La condition `StringEquals` nécessite que `ExampleApp` soit capitalisé tel qu'il est spécifié.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Appname": "ExampleApp"
    }
  }
}
```

Pour exiger une clé contextuelle de chiffrement distinguant majuscules et minuscules, utilisez la condition kms : EncryptionContextKeys policy avec un opérateur de condition sensible aux

majuscules et minuscules, tel que. `StringEquals` Dans cette condition de politique, étant donné que la clé de contexte de chiffrement est la valeur de cette condition de politique, sa sensibilité à la casse est déterminée par l'opérateur de condition.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "ForAnyValue:StringEquals": {
      "kms:EncryptionContextKeys": "AppName"
    }
  }
}
```

Pour exiger une évaluation sensible aux majuscules et minuscules de la clé de contexte de chiffrement et de sa valeur, utilisez les conditions de politique de clé de contexte *`kms:EncryptionContextKeys`* et kms: EncryptionContext : ensemble dans la même déclaration de politique. L'opérateur de condition sensible à la casse (tel que `StringEquals`) s'applique toujours à la valeur de la condition. La clé de contexte de chiffrement (telle que AppName) est la valeur de la condition `kms:EncryptionContextKeys`. La valeur du contexte de chiffrement (telle que`ExampleApp`) est la valeur de la condition kms: EncryptionContext : clé de contexte.

Par exemple, dans l'exemple suivant d'instruction de politique de clé, étant donné que l'opérateur `StringEquals` est sensible à la casse, la clé de contexte de chiffrement et la valeur de contexte de chiffrement sont toutes deux sensibles à la casse.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "ForAnyValue:StringEquals": {
      "kms:EncryptionContextKeys": "AppName"
    },
```

```
    "StringEquals": {
      "kms:EncryptionContext:AppName": "ExampleApp"
    }
  }
}
```

Utilisation de variables dans une condition de contexte de chiffrement

La clé et la valeur d'une paire de contexte de chiffrement doivent être des chaînes littérales simples. Il ne peut pas s'agir d'entiers ou d'objets, ou d'un type qui n'est pas entièrement résolu. Si vous utilisez un autre type, tel qu'un entier ou un flottant, il est AWS KMS interprété comme une chaîne littérale.

```
"encryptionContext": {
    "department": "10103.0"
}
```

Toutefois, la valeur de la clé de condition `kms:EncryptionContext:`*`context-key`* peut être une variable de politique IAM. Ces variables de politique sont résolues lors de l'exécution en fonction des valeurs de la demande. Par exemple, `aws:CurrentTime` est résolue à l'heure de la demande et `aws:username` est résolue au nom convivial de l'appelant.

Vous pouvez utiliser ces variables de politique pour créer une instruction de politique avec une condition qui nécessite des informations très spécifiques dans un contexte de chiffrement, comme le nom d'utilisateur de l'appelant. Comme elle contient une variable, vous pouvez utiliser la même instruction de politique pour tous les utilisateurs qui peuvent assumer le rôle. Vous n'avez pas besoin d'écrire une instruction de politique distincte pour chaque utilisateur.

Prenons un exemple : vous voulez que tous les utilisateurs qui peuvent endosser un rôle utilisent la même clé KMS pour chiffrer et déchiffrer leurs données. Cependant, vous souhaitez leur permettre de déchiffrer uniquement les données qu'ils ont chiffrées. Commencez par exiger que chaque demande AWS KMS inclue un contexte de chiffrement dans lequel la clé est le nom d'utilisateur de l'appelant `user` et dont la valeur est le nom AWS d'utilisateur, tel que le suivant.

```
"encryptionContext": {
    "user": "bob"
}
```

Ensuite, pour appliquer cette exigence, vous pouvez utiliser une instruction de politique comme celle de l'exemple suivant. Cette instruction de politique accorde au rôle `TestTeam` l'autorisation de

chiffrer et de déchiffrer les données avec la clé KMS. Toutefois, l'autorisation est uniquement valable lorsque le contexte de chiffrement de la demande inclut une paire `"user": "<username>"`. Pour représenter le nom d'utilisateur, la condition utilise la variable de politique `aws:username`.

Lorsque la demande est évaluée, le nom d'utilisateur de l'appelant remplace la variable dans la condition. Ainsi, la condition nécessite un contexte de chiffrement `"user": "bob"` pour « bob » et `"user": "alice"` pour « alice ».

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/TestTeam"
  },
  "Action": [
    "kms:Decrypt",
    "kms:Encrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:user": "${aws:username}"
    }
  }
}
```

Vous pouvez utiliser une variable de politique IAM uniquement dans la valeur de la clé de condition `kms:EncryptionContext:context-key`. Vous ne pouvez pas utiliser une variable dans la clé.

Vous pouvez également utiliser des clés de contexte spécifiques au fournisseur dans des variables. Ces clés contextuelles identifient de manière unique les utilisateurs qui se sont connectés à AWS l'aide de la fédération d'identité Web.

Comme toutes les variables, celles-ci peuvent être utilisées uniquement dans la condition de politique `kms:EncryptionContext:context-key`, et non dans le contexte de chiffrement réel. Et elles peuvent être utilisées uniquement dans la valeur de la condition, pas dans la clé.

Par exemple, l'instruction de politique de clé suivante est similaire à la précédente. Toutefois, la condition nécessite un contexte de chiffrement où la clé est `sub` et la valeur identifie de façon unique un utilisateur connecté à un groupe d'utilisateurs Amazon Cognito. Pour plus de détails sur l'identification des utilisateurs et les rôles dans Amazon Cognito, veuillez consulter Rôles IAM dans le guide du développeur Amazon Cognito.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/TestTeam"
  },
  "Action": [
    "kms:Decrypt",
    "kms:Encrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
       "kms:EncryptionContext:sub": "${cognito-identity.amazonaws.com:sub}"
    }
  }
}
```

Voir aussi

- the section called “km : EncryptionContextKeys”
- the section called “km : GrantConstraintType”

km : EncryptionContextKeys

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d’API	Type de stratégie

`kms:EncryptionContextKeys`	Chaîne (liste)	Multi-valeurs	`CreateGrant`
			`Decrypt`
			`Encrypt`
			`GenerateDataKey`
			`GenerateDataKeyPair`
				Politiques de clé et politiques IAM

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie

			`GenerateDataKeyPairWithoutPlaintext`
			
			`GenerateDataKeyWithoutPlaintext`
			
			`ReEncrypt`
			
			`RetireGrant`	

Vous pouvez utiliser la clé de condition `kms:EncryptionContextKeys` pour contrôler l'accès à une clé KMS de chiffrement symétrique en fonction du contexte de chiffrement d'une demande d'opération de chiffrement. Utilisez cette clé de condition pour évaluer uniquement la clé dans chaque paire de contexte de chiffrement. Utilisez la clé de condition `kms:EncryptionContext:`*`context-key`* afin d'évaluer à la fois la clé et la valeur dans le contexte de chiffrement.

Vous ne pouvez pas spécifier de contexte de chiffrement dans une opération de chiffrement avec une clé KMS asymétrique ou une clé KMS HMAC. Les algorithmes asymétriques et les algorithmes MAC ne prennent pas en charge un contexte de chiffrement.

> **Note**
>
> Les valeurs des clés de condition, y compris une clé de contexte de chiffrement, doivent être conformes aux règles de caractères et de codage des politiques AWS KMS clés. Vous ne pourrez peut-être pas utiliser cette clé de condition pour exprimer toutes les clés du contexte de chiffrement valides. Pour plus d'informations sur les règles de document de politique de clé, voir Format de politique de clé. Pour plus d'informations sur les règles du document de politique IAM, veuillez consulter Exigences relatives aux noms IAM dans le Guide de l'utilisateur IAM.

Il s'agit d'une clé de condition multi-valeurs. Vous pouvez spécifier plusieurs paires de contexte de chiffrement dans chaque demande d'API. `kms:EncryptionContextKeys` compare les clés de contexte de chiffrement dans la demande à l'ensemble des clés de contexte de chiffrement dans la politique. Pour déterminer comment ces ensembles sont comparés, vous devez fournir un opérateur d'ensemble `ForAnyValue` ou `ForAllValues` dans la condition de politique. Pour plus de détails sur les opérateurs d'ensemble, veuillez consulter Utilisation de plusieurs clés et valeurs dans le Guide de l'utilisateur IAM.

- `ForAnyValue` : au moins une clé de contexte de chiffrement dans la demande doit correspondre à une clé de contexte de chiffrement dans la condition de politique. D'autres clés de contexte de chiffrement sont autorisées. Si la demande n'a aucun contexte de chiffrement, la condition n'est pas remplie.
- `ForAllValues` : chaque clé de contexte de chiffrement de la demande doit correspondre à une clé de contexte de chiffrement dans la condition de politique. Cet opérateur d'ensemble limite les clés de contexte de chiffrement à celles de la condition de politique. Il ne nécessite aucune clé de contexte de chiffrement, mais il interdit les clés de contexte de chiffrement non spécifiées.

L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:EncryptionContextKeys` avec l'opérateur d'ensemble `ForAnyValue`. Cette instruction de politique autorise l'utilisation d'une clé KMS pour les opérations spécifiées, mais uniquement si au moins une des paires de contexte de chiffrement de la demande inclut la clé `AppName`, peu importe sa valeur.

Par exemple, cette instruction de politique de clé autorise une demande `GenerateDataKey` avec deux paires de contexte de chiffrement, `AppName=Helper` et `Project=Alpha`, car la première paire de contexte de chiffrement répond à la condition. Une demande avec uniquement `Project=Alpha` ou sans contexte de chiffrement échouerait.

Comme l'opération StringEqualsconditionnelle distingue les majuscules et minuscules, cette déclaration de politique requiert l'orthographe et les majuscules de la clé contextuelle de chiffrement. Toutefois, vous pouvez utiliser un opérateur de condition qui ignore la casse de la clé, par exemple `StringEqualsIgnoreCase`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
```

```
  "Action": [
    "kms:Encrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition": {
    "ForAnyValue:StringEquals": {
      "kms:EncryptionContextKeys": "AppName"
    }
  }
}
```

Vous pouvez également utiliser la clé de condition `kms:EncryptionContextKeys` pour exiger un contexte de chiffrement (tout contexte de chiffrement) dans les opérations de cryptographiques qui utilisent la clé KMS.

L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:EncryptionContextKeys` avec l'opérateur de condition Null pour autoriser l'accès à une clé KMS uniquement lorsque le contexte de chiffrement de la demande d'API n'est pas nul. Cette condition ne vérifie pas les clés ou les valeurs du contexte de chiffrement. Elle vérifie uniquement que le contexte de chiffrement existe.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": [
    "kms:Encrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:EncryptionContextKeys": false
    }
  }
}
```

Voir aussi

- kms EncryptionContext : touche contextuelle

- km : GrantConstraintType

km : ExpirationModel

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ExpirationModel`	Chaîne	À valeur unique	`ImportKeyMaterial`	Politiques de clé et politiques IAM

La clé de `kms:ExpirationModel` condition contrôle l'accès à l'ImportKeyMaterialopération en fonction de la valeur du ExpirationModelparamètre dans la demande.

`ExpirationModel` est un paramètre facultatif qui détermine si les éléments de clé importés arrivent à expiration. Les valeurs valides sont `KEY_MATERIAL_EXPIRES` et `KEY_MATERIAL_DOES_NOT_EXPIRE`. La valeur par défaut est `KEY_MATERIAL_EXPIRES`.

La date et l'heure d'expiration sont déterminées par la valeur du ValidToparamètre. Le paramètre `ValidTo` est obligatoire, sauf si la valeur du paramètre `ExpirationModel` est `KEY_MATERIAL_DOES_NOT_EXPIRE`. Vous pouvez également utiliser la clé de ValidTo condition kms : pour exiger une date d'expiration particulière comme condition d'accès.

L'exemple d'instruction de politique suivant utilise la clé de condition `kms:ExpirationModel` pour autoriser les utilisateurs à importer les éléments de clé dans une clé KMS uniquement lorsque la demande inclut le paramètre `ExpirationModel` et que sa valeur est `KEY_MATERIAL_DOES_NOT_EXPIRE`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:ImportKeyMaterial",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ExpirationModel": "KEY_MATERIAL_DOES_NOT_EXPIRE"
    }
```

```
  }
}
```

Vous pouvez également utiliser la clé de condition `kms:ExpirationModel` pour autoriser les utilisateurs à importer les éléments de clé uniquement lorsque ceux-ci expirent. L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:ExpirationModel` avec l'opérateur de condition Null pour autoriser les utilisateurs à importer les éléments de clé uniquement lorsque la demande ne dispose pas de paramètre `ExpirationModel`. La valeur par défaut pour ExpirationModel est`KEY_MATERIAL_EXPIRES`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:ImportKeyMaterial",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:ExpirationModel": true
    }
  }
}
```

Voir aussi

- km : ValidTo
- km : WrappingAlgorithm
- km : WrappingKeySpec

km : GrantConstraintType

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:GrantConstraintType`	Chaîne	À valeur unique	`CreateGrant` `RetireGrant`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'CreateGrantopération en fonction du type de contrainte d'autorisation figurant dans la demande.

Lorsque vous créez un octroi, vous pouvez éventuellement spécifier une contrainte d'octroi pour autoriser les opérations permises par l'octroi seulement lorsqu'un contexte de chiffrement particulier est présent. La contrainte d'octroi peut être de deux types : `EncryptionContextEquals` ou `EncryptionContextSubset`. Vous pouvez utiliser cette clé de condition pour vérifier que la demande contient un type ou l'autre.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:GrantConstraintType` pour autoriser les utilisateurs à créer des octrois uniquement lorsque la demande inclut une contrainte d'octroi `EncryptionContextEquals`. L'exemple suivant montre une instruction de politique dans une politique de clé.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:GrantConstraintType": "EncryptionContextEquals"
    }
  }
}
```

Voir aussi

- kms EncryptionContext : touche contextuelle
- km : EncryptionContextKeys
- km : GrantIsFor AWSResource

- km : GrantOperations
- km : GranteePrincipal
- km : RetiringPrincipal

km : GrantIsFor AWSResource

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie

`kms:GrantIsForAWSResource`	Booléen	À valeur unique	`CreateGrant`
			`ListGrants`
			`RevokeGrant`
				Politiques de clé et politiques IAM

Autorise ou refuse l'autorisation pour les RevokeGrantopérations CreateGrantListGrants, ou uniquement lorsqu'un AWS service intégré AWS KMS appelle l'opération au nom de l'utilisateur. Cette condition de politique ne permet pas à l'utilisateur d'appeler ces opérations d'octroi directement.

L'exemple suivant d'instruction de politique de clé utilise la clé de condition `kms:GrantIsForAWSResource`. Il permet aux AWS services intégrés AWS KMS, tels qu'Amazon EBS, de créer des subventions sur cette clé KMS au nom du principal spécifié.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:GrantIsForAWSResource": true
    }
  }
}
```

Voir aussi

- km : GrantConstraintType
- km : GrantOperations
- km : GranteePrincipal
- km : RetiringPrincipal

km : GrantOperations

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:GrantOperations`	Chaîne	Multi-valeurs	`CreateGrant`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'CreateGrantopération en fonction des opérations d'autorisation figurant dans la demande. Par exemple, vous pouvez autoriser les utilisateurs à créer des octrois qui délèguent l'autorisation de chiffrer mais pas de déchiffrer. Pour plus d'informations sur les octrois, veuillez consulter Utilisation d'octrois.

Il s'agit d'une clé de condition multi-valeurs. `kms:GrantOperations` compare l'ensemble d'opérations d'octrois dans la demande `CreateGrant` à l'ensemble d'opérations d'octrois dans la politique. Pour déterminer comment ces ensembles sont comparés, vous devez fournir un opérateur d'ensemble `ForAnyValue` ou `ForAllValues` dans la condition de politique. Pour plus de détails sur les opérateurs d'ensemble, veuillez consulter Utilisation de plusieurs clés et valeurs dans le Guide de l'utilisateur IAM.

- `ForAnyValue` : au moins une opération d'octroi dans la demande doit correspondre à l'une des opérations d'octrois dans la condition de politique. D'autres opérations d'octrois sont autorisées.
- ForAllValues: Chaque opération de subvention figurant dans la demande doit correspondre à une opération de subvention figurant dans la condition de politique. Cet opérateur d'ensemble limite les opérations d'octrois à celles spécifiées dans la condition de politique. Il ne nécessite aucune opération d'octroi, mais il interdit les opérations d'octrois non spécifiées.

  ForAllValues renvoie également true lorsqu'il n'y a aucune opération de subvention dans la demande, mais `CreateGrant` ne l'autorise pas. Si le paramètre `Operations` manque ou qu'il a une valeur nulle, la demande `CreateGrant` échoue.

L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:GrantOperations` pour créer des octrois uniquement lorsque les opérations d'octrois sont `Encrypt`, `ReEncryptTo` ou les deux. Si l'octroi inclut toute autre opération, la demande `CreateGrant` échoue.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "ForAllValues:StringEquals": {
      "kms:GrantOperations": [
        "Encrypt",
        "ReEncryptTo"
      ]
    }
  }
}
```

Si vous changez l'opérateur d'ensemble dans la condition de politique en `ForAnyValue`, l'instruction de politique exigerait qu'au moins une des opérations d'octrois dans l'octroi soit `Encrypt` ou `ReEncryptTo`, mais elle autoriserait d'autres opérations d'octrois, telles que `Decrypt` ou `ReEncryptFrom`.

Voir aussi

- km : GrantConstraintType
- km : GrantIsFor AWSResource
- km : GranteePrincipal
- km : RetiringPrincipal

km : GranteePrincipal

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:GranteePrincipal`	Chaîne	À valeur unique	`CreateGrant`	Politiques de clé et IAM

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'CreateGrantopération en fonction de la valeur du GranteePrincipalparamètre dans la demande. Par exemple, vous pouvez créer des octrois pour utiliser une clé KMS uniquement lorsque le principal bénéficiaire de la demande `CreateGrant` correspond au principal spécifié dans l'instruction de condition.

Pour spécifier le principal du bénéficiaire, utilisez le Amazon Resource Name (ARN) d'un AWS principal. Les principaux valides incluent les utilisateurs IAM Comptes AWS, les rôles IAM, les utilisateurs fédérés et les utilisateurs des rôles assumés. Pour obtenir de l'aide sur la syntaxe ARN d'un principal, consultez IAM ARNs dans le guide de l'utilisateur IAM.

L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:GranteePrincipal` afin de créer des octrois pour une clé KMS uniquement lorsque le principal bénéficiaire de l'octroi est le `LimitedAdminRole`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:GranteePrincipal": "arn:aws:iam::111122223333:role/LimitedAdminRole"
    }
  }
}
```

Voir aussi

- km : GrantConstraintType
- km : GrantIsFor AWSResource
- km : GrantOperations
- km : RetiringPrincipal

km : KeyAgreementAlgorithm

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:KeyAgreementAlgorithm`	Chaîne	À valeur unique	`DeriveSharedSecret`	Politiques de clé et politiques IAM

Vous pouvez utiliser la clé de `kms:KeyAgreementAlgorithm` condition pour contrôler l'accès à l'DeriveSharedSecretopération en fonction de la valeur du `KeyAgreementAlgorithm` paramètre dans la demande. La seule valeur valide pour `KeyAgreementAlgorithm` est`ECDH`.

Par exemple, la déclaration de politique clé suivante utilise la clé de `kms:KeyAgreementAlgorithm` condition pour refuser tout accès à DeriveSharedSecret moins que ce ne `KeyAgreementAlgorithm` soit le cas`ECDH`.

```
{
       "Effect": "Deny",
       "Principal": {
         "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
       },
       "Action": "kms:DeriveSharedSecret",
       "Resource": "*",
       "Condition": {
            "StringNotEquals": {
               "kms:KeyAgreementAlgorithm": "ECDH"
         }
       }
}
```

Voir aussi

- the section called “km : KeyUsage”

km : KeyOrigin

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d’API	Type de stratégie
`kms:KeyOrigin`	Chaîne	À valeur unique	`CreateKey` Opérations liées aux ressources de clé KMS	Politiques IAM Politiques de clé et politiques IAM

La clé de condition `kms:KeyOrigin` contrôle l'accès aux opérations en fonction de la valeur de la propriété `Origin` de la clé KMS créée par l'opération ou utilisée dans cette dernière. Elle fonctionne comme une condition de ressource ou une condition de demande.

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'CreateKeyopération en fonction de la valeur du paramètre Origin dans la demande. Les valeurs valides pour `Origin` sont `AWS_KMS`, `AWS_CLOUDHSM` et `EXTERNAL`.

Par exemple, vous pouvez créer une clé KMS uniquement lorsque le contenu clé est généré dans AWS KMS (`AWS_KMS`), uniquement lorsque le matériau clé est généré dans un AWS CloudHSM cluster associé à un magasin de clés personnalisé (`AWS_CLOUDHSM`), ou uniquement lorsque le matériau clé est importé depuis une source externe (`EXTERNAL`).

L'exemple de déclaration de politique clé suivant utilise la clé de `kms:KeyOrigin` condition pour créer une clé KMS uniquement lors de la AWS KMS création du matériel clé.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
      },
      "Action": "kms:CreateKey",
      "Resource": "*",
```

```
      "Condition": {
        "StringEquals": {
          "kms:KeyOrigin": "AWS_KMS"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:GenerateDataKeyPair",
        "kms:GenerateDataKeyPairWithoutPlaintext",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "kms:KeyOrigin": "AWS_CLOUDHSM"
        }
      }
    }
  ]
}
```

Vous pouvez également utiliser la clé de condition `kms:KeyOrigin` pour contrôler l'accès aux opérations qui utilisent ou gèrent une clé KMS en fonction de la propriété `Origin` de la clé KMS utilisée pour l'opération. L'opération doit être une opération de ressource de clé KMS, c'est-à-dire une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS key` dans la colonne `Resources` de l'opération.

Par exemple, la politique IAM suivante permet aux principaux d'effectuer les opérations de ressources de clé KMS spécifiées, mais uniquement avec les clés KMS du compte qui ont été créées dans un magasin de clés personnalisé.

```
{
```

```
  "Effect": "Allow",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:GenerateDataKey",
    "kms:GenerateDataKeyWithoutPlaintext",
    "kms:GenerateDataKeyPair",
    "kms:GenerateDataKeyPairWithoutPlaintext",
    "kms:ReEncrypt*"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "AWS_CLOUDHSM"
    }
  }
}
```

Voir aussi

- km : BypassPolicyLockoutSafetyCheck
- km : KeySpec
- km : KeyUsage

km : KeySpec

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:KeySpec`	Chaîne	À valeur unique	`CreateKey` Opérations liées aux ressources de clé KMS	Politiques IAM Politiques de clé et politiques IAM

La clé de condition `kms:KeySpec` contrôle l'accès aux opérations en fonction de la valeur de la propriété `KeySpec` de la clé KMS créée par l'opération ou utilisée dans cette dernière.

Vous pouvez utiliser cette clé de condition dans une politique IAM pour contrôler l'accès à l'CreateKeyopération en fonction de la valeur du KeySpecparamètre dans une `CreateKey` demande. Par exemple, vous pouvez utiliser cette condition pour autoriser les utilisateurs à créer uniquement des clés KMS de chiffrement symétriques ou des clés KMS HMAC.

L'exemple d'instruction de politique IAM suivant utilise la clé de condition `kms:KeySpec` pour autoriser les principaux à créer des clés KMS asymétriques RSA uniquement. L'autorisation n'est valide que lorsque la valeur `KeySpec` dans la demande commence par `RSA_`.

```
{
  "Effect": "Allow",
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:KeySpec": "RSA_*"
    }
  }
}
```

Vous pouvez également utiliser la clé de condition `kms:KeySpec` pour contrôler l'accès aux opérations qui utilisent ou gèrent une clé KMS en fonction de la propriété `KeySpec` de la clé KMS utilisée pour l'opération. L'opération doit être une opération de ressource de clé KMS, c'est-à-dire une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS key` dans la colonne `Resources` de l'opération.

Par exemple, la politique IAM suivante permet aux principaux d'effectuer les opérations de ressources de clé KMS spécifiées, mais uniquement avec les clés KMS de chiffrement symétriques du compte.

```
{
  "Effect": "Allow",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:DescribeKey"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
```

```
  "Condition": {
    "StringEquals": {
      "kms:KeySpec": "SYMMETRIC_DEFAULT"
    }
  }
}
```

Voir aussi

- km : BypassPolicyLockoutSafetyCheck
- kms : CustomerMasterKeySpec (obsolète)
- km : DataKeyPairSpec
- km : KeyOrigin
- km : KeyUsage

km : KeyUsage

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:KeyUsage`	Chaîne	À valeur unique	`CreateKey` Opérations liées aux ressources de clé KMS	Politiques IAM Politiques de clé et politiques IAM

La clé de condition `kms:KeyUsage` contrôle l'accès aux opérations en fonction de la valeur de la propriété `KeyUsage` de la clé KMS créée par l'opération ou utilisée dans cette dernière.

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'CreateKeyopération en fonction de la valeur du KeyUsageparamètre dans la demande. Les valeurs valides pour `KeyUsage` sont `ENCRYPT_DECRYPTSIGN_VERIFY`,`GENERATE_VERIFY_MAC`, et`KEY_AGREEMENT`.

Par exemple, vous pouvez créer une clé KMS uniquement lorsque `KeyUsage` a pour valeur `ENCRYPT_DECRYPT` ou refuser à un utilisateur cette autorisation lorsque `KeyUsage` a pour valeur `SIGN_VERIFY`.

L'exemple d'instruction de politique IAM suivant utilise la clé de condition `kms:KeyUsage` pour créer une clé KMS uniquement lorsque le paramètre `KeyUsage` a pour valeur `ENCRYPT_DECRYPT`.

```
{
  "Effect": "Allow",
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:KeyUsage": "ENCRYPT_DECRYPT"
    }
  }
}
```

Vous pouvez également utiliser la clé de condition `kms:KeyUsage` pour contrôler l'accès aux opérations qui utilisent ou gèrent une clé KMS en fonction de la propriété `KeyUsage` de la clé KMS utilisée pour l'opération. L'opération doit être une opération de ressource de clé KMS, c'est-à-dire une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS key` dans la colonne `Resources` de l'opération.

Par exemple, la politique IAM suivante permet aux principaux d'effectuer les opérations de ressources de clé KMS spécifiées, mais uniquement avec les clés KMS du compte qui sont utilisées pour la signature et la vérification.

```
{
  "Effect": "Allow",
  "Action": [
    "kms:CreateGrant",
    "kms:DescribeKey",
    "kms:GetPublicKey",
    "kms:ScheduleKeyDeletion"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:KeyUsage": "SIGN_VERIFY"
    }
  }
}
```

Voir aussi

- km : BypassPolicyLockoutSafetyCheck
- kms : CustomerMasterKeyUsage (obsolète)
- km : KeyOrigin
- km : KeySpec

km : MacAlgorithm

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:MacAlgorithm`	Chaîne	À valeur unique	`GenerateMac` `VerifyMac`	Politiques de clé et politiques IAM

Vous pouvez utiliser la clé de `kms:MacAlgorithm` condition pour contrôler l'accès aux VerifyMacopérations GenerateMacet en fonction de la valeur du `MacAlgorithm` paramètre dans la demande.

L'exemple de politique de clé suivant permet aux utilisateurs qui peuvent endosser le rôle `testers` d'utiliser la clé KMS HMAC pour générer et vérifier les balises HMAC uniquement lorsque l'algorithme MAC de la requête est `HMAC_SHA_384` ou `HMAC_SHA_512`. Cette politique utilise deux instructions de politique distinctes ayant chacune leur propre condition. Si vous spécifiez plusieurs algorithmes MAC dans une seule instruction de condition, la condition nécessite les deux algorithmes, au lieu de l'un ou de l'autre.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/testers"
      },
      "Action": [
        "kms:GenerateMac",
```

```
        "kms:VerifyMac"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:MacAlgorithm": "HMAC_SHA_384"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/testers"
      },
      "Action": [
        "kms:GenerateMac",
        "kms:VerifyMac"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:MacAlgorithm": "HMAC_SHA_512"
        }
      }
    }
  ]
}
```

Voir aussi

- the section called “km : EncryptionAlgorithm”
- km : SigningAlgorithm

km : MessageType

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d’API	Type de stratégie
`kms:MessageType`	Chaîne	À valeur unique	`Sign` `Verify`	Politiques de clé et politiques IAM

La clé de condition `kms:MessageType` contrôle l'accès aux opérations Sign et Verify en fonction de la valeur du paramètre `MessageType` de la demande. Les valeurs valides pour `MessageType` sont `RAW` et `DIGEST`.

Par exemple, l'instruction de politique de clé suivante utilise la clé de condition `kms:MessageType` afin d'utiliser une clé KMS asymétrique pour signer un message, mais pas un condensé du message.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:Sign",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:MessageType": "RAW"
    }
  }
}
```

Voir aussi

- the section called “km : SigningAlgorithm”

km : MultiRegion

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d’API	Type de stratégie
`kms:MultiRegion`	Booléen	À valeur unique	`CreateKey` Opérations liées aux ressources de clé KMS	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour autoriser les opérations uniquement sur des clés à région unique ou uniquement sur des clés multi-région. La clé de `kms:MultiRegion` condition contrôle l'accès aux AWS KMS opérations sur les clés KMS et à l'CreateKeyopération en fonction de

la valeur de la `MultiRegion` propriété de la clé KMS. Les valeurs valides sont `true` (multi-région) et `false` (région unique). Toutes les clés KMS ont une propriété `MultiRegion`.

Par exemple, l'instruction de politique IAM suivante utilise la clé de condition `kms:MultiRegion` afin d'autoriser les principaux à créer uniquement des clés à région unique.

```
{
  "Effect": "Allow",
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:MultiRegion": false
    }
  }
}
```

km : MultiRegionKeyType

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:MultiRegionKeyType`	Chaîne	À valeur unique	`CreateKey` Opérations liées aux ressources de clé KMS	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour autoriser les opérations uniquement sur des clés primaires multi-région ou uniquement sur des clés de réplica multi-région. La clé de `kms:MultiRegionKeyType` condition contrôle l'accès aux AWS KMS opérations sur les clés KMS et l'CreateKeyopération en fonction de la `MultiRegionKeyType` propriété de la clé KMS. Les valeurs valides sont `PRIMARY` et `REPLICA`. Seules les clés multi-région ont une propriété `MultiRegionKeyType`.

En général, vous utilisez la clé de condition `kms:MultiRegionKeyType` dans une politique IAM pour contrôler l'accès à plusieurs clés KMS. Toutefois, comme une clé multi-région donnée peut se changer en primaire ou en réplica, vous devrez peut-être utiliser cette condition dans une politique

de clé pour autoriser une opération uniquement lorsque la clé multi-région particulière est une clé primaire ou réplica.

Par exemple, l'instruction de politique IAM suivante utilise la clé de condition `kms:MultiRegionKeyType` pour permettre aux principaux de planifier et d'annuler la suppression des clés uniquement sur les clés de réplica multi-région dans le Compte AWS spécifié.

```
{
  "Effect": "Allow",
  "Action": [
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "arn:aws:kms:*:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:MultiRegionKeyType": "REPLICA"
    }
  }
}
```

Pour autoriser ou refuser l'accès à toutes les clés multi-région, vous pouvez utiliser les deux valeurs ou une valeur nulle avec `kms:MultiRegionKeyType`. Cependant, la clé de MultiRegion condition kms : est recommandée à cette fin.

km : PrimaryRegion

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:PrimaryRegion`	Chaîne (liste)	À valeur unique	`UpdatePrimaryRegion`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour limiter les régions de destination dans une UpdatePrimaryRegionopération. Ce sont eux Régions AWS qui peuvent héberger vos clés primaires multirégionales.

La touche de `kms:PrimaryRegion` condition contrôle l'accès à l'UpdatePrimaryRegionopération en fonction de la valeur du `PrimaryRegion` paramètre. Le `PrimaryRegion` paramètre spécifie la clé

Région AWS de réplique multirégionale qui est promue en clé principale. La valeur de la condition est un ou plusieurs Région AWS noms, tels que `us-east-1` ou`ap-southeast-2`, ou des modèles de noms de région, tels que `eu-*`

Par exemple, l'instruction de politique de clé suivante utilise la clé de condition `kms:PrimaryRegion` afin de permettre aux principaux de mettre à jour la région primaire d'une clé multi-région vers une des quatre régions spécifiées.

```
{
  "Effect": "Allow",
  "Action": "kms:UpdatePrimaryRegion",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Developer"
  },
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:PrimaryRegion": [
         "us-east-1",
         "us-west-2",
         "eu-west-3",
         "ap-southeast-2"
      ]
    }
  }
}
```

km : ReEncryptOnSameKey

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ReEncryptOnSameKey`	Booléen	À valeur unique	`ReEncrypt`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'ReEncryptopération selon que la demande spécifie ou non une clé KMS de destination identique à celle utilisée pour le chiffrement d'origine.

Par exemple, l'instruction de politique de clé suivante utilise la clé de condition `kms:ReEncryptOnSameKey` pour rechiffrer uniquement lorsque la clé KMS de destination est identique à celle utilisée pour le chiffrement d'origine.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:ReEncrypt*",
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:ReEncryptOnSameKey": true
    }
  }
}
```

km : RequestAlias

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:RequestAlias`	Chaîne (liste)	À valeur unique	Opérations cryptographiques DescribeKey GetPublicKey	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour autoriser une opération uniquement lorsque la demande utilise un alias particulier pour identifier la clé KMS. La clé de condition `kms:RequestAlias` contrôle l'accès à une clé KMS utilisée lors d'une opération cryptographique, `GetPublicKey`, ou `DescribeKey` en fonction de l'alias qui identifie cette clé KMS dans la demande. (Cette condition de politique n'a aucun effet sur l'GenerateRandomopération car celle-ci n'utilise pas de clé ou d'alias KMS.)

Cette condition prend en charge le contrôle d'accès basé sur les attributs (ABAC) dans AWS KMS, qui vous permet de contrôler l'accès aux clés KMS en fonction des balises et des alias d'une clé

KMS. Vous pouvez utiliser des balises et des alias pour autoriser ou refuser l'accès à une clé KMS sans modifier les politiques ou les octrois. Pour plus de détails, veuillez consulter ABAC pour AWS KMS.

Pour spécifier l'alias dans cette condition de politique, utilisez un nom d'alias, tel que `alias/project-alpha`, ou un modèle de nom d'alias, tel que `alias/*test*`. Vous ne pouvez pas spécifier un ARN d'alias dans la valeur de cette clé de condition.

Pour satisfaire à cette condition, la valeur du paramètre `KeyId` dans la demande doit être un nom d'alias ou un ARN d'alias correspondant. Si la demande utilise un identifiant de clé, elle ne satisfait pas à la condition, même si elle identifie la même clé KMS.

Par exemple, la déclaration de politique clé suivante permet au principal d'appeler l'GenerateDataKeyopération sur la clé KMS. Cependant, cela n'est autorisé que lorsque la valeur du paramètre `KeyId` dans la demande est `alias/finance-key` ou un ARN d'alias avec ce nom d'alias, tel que `arn:aws:kms:us-west-2:111122223333:alias/finance-key`.

```
{
  "Sid": "Key policy using a request alias condition",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/developer"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:RequestAlias": "alias/finance-key"
    }
  }
}
```

Vous ne pouvez pas utiliser cette clé de condition pour contrôler l'accès aux opérations d'alias, telles que CreateAliasou DeleteAlias. Pour de plus amples informations sur le contrôle de l'accès aux opérations d'alias, veuillez consulter Contrôle de l'accès aux alias.

km : ResourceAliases

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ResourceAliases`	Chaîne (liste)	Multi-valeurs	Opérations liées aux ressources de clé KMS	Politiques IAM uniquement

Utilisez cette clé de condition pour contrôler l'accès à une clé KMS en fonction des alias associés à la clé KMS. L'opération doit être une opération de ressource de clé KMS, c'est-à-dire une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS key` dans la colonne `Resources` de l'opération.

Cette condition prend en charge le contrôle d'accès basé sur les attributs (ABAC) dans AWS KMS. Avec l'ABAC, vous pouvez contrôler l'accès aux clés KMS en fonction des balises attribuées à une clé KMS et des alias associés à une clé KMS. Vous pouvez utiliser des balises et des alias pour autoriser ou refuser l'accès à une clé KMS sans modifier les politiques ou les octrois. Pour plus de détails, veuillez consulter ABAC pour AWS KMS.

Un alias doit être unique dans une région Compte AWS et, mais cette condition vous permet de contrôler l'accès à plusieurs clés KMS dans la même région (à l'aide de l'opérateur Régions AWS de `StringLike` comparaison) ou à plusieurs clés KMS dans différents comptes.

> **Note**
>
> La ResourceAliases condition kms : n'est effective que lorsque la clé KMS est conforme aux alias par quota de clé KMS. Si une clé KMS dépasse ce quota, les principaux autorisés à utiliser la clé KMS par la condition `kms:ResourceAliases` se voient refuser l'accès à la clé KMS.

Pour spécifier l'alias dans cette condition de politique, utilisez un nom d'alias, tel que `alias/project-alpha`, ou un modèle de nom d'alias, tel que `alias/*test*`. Vous ne pouvez pas spécifier un ARN d'alias dans la valeur de cette clé de condition. Pour satisfaire à cette condition, la

clé KMS utilisée dans l'opération doit avoir l'alias spécifié. Peu importe si ou comment la clé KMS est identifiée dans la demande pour l'opération.

Il s'agit d'une clé de condition multi-valeurs qui compare l'ensemble d'alias associé à une clé KMS à l'ensemble d'alias de la politique. Pour déterminer comment ces ensembles sont comparés, vous devez fournir un opérateur d'ensemble `ForAnyValue` ou `ForAllValues` dans la condition de politique. Pour plus de détails sur les opérateurs d'ensemble, veuillez consulter Utilisation de plusieurs clés et valeurs dans le Guide de l'utilisateur IAM.

- ForAnyValue: Au moins un alias associé à la clé KMS doit correspondre à un alias figurant dans la condition de politique. D'autres alias sont autorisés. Si la clé KMS n'a pas d'alias, la condition n'est pas remplie.
- ForAllValues: Chaque alias associé à la clé KMS doit correspondre à un alias indiqué dans la politique. Cet opérateur d'ensemble limite les alias associés à la clé KMS à ceux de la condition de politique. Il ne nécessite aucun alias, mais il interdit les alias non spécifiés.

Par exemple, la déclaration de politique IAM suivante permet au principal d'appeler l'GenerateDataKeyopération sur n'importe quelle clé KMS spécifiée Compte AWS associée à l'`finance-key`alias. (Les politiques de clé des clés KMS affectées doivent également permettre au compte du principal de les utiliser pour cette opération.) Pour indiquer que la condition est remplie lorsque l'un des nombreux alias qui peuvent être associés à la clé KMS est `alias/finance-key`, la condition utilise l'opérateur d'ensemble `ForAnyValue`.

Puisque la condition `kms:ResourceAliases` est basée sur la ressource et non pas sur la demande, un appel vers `GenerateDataKey` réussit pour toute clé KMS associée à l'alias `finance-key`, même si la demande utilise un ID de clé ou un ARN de clé pour identifier la clé KMS.

```
{
  "Sid": "AliasBasedIAMPolicy",
  "Effect": "Allow",
  "Action": "kms:GenerateDataKey",
  "Resource": [
    "arn:aws:kms:*:111122223333:key/*",
    "arn:aws:kms:*:444455556666:key/*"
  ],
  "Condition": {
    "ForAnyValue:StringEquals": {
      "kms:ResourceAliases": "alias/finance-key"
    }
```

```
  }
}
```

L'exemple suivant de politique IAM permet au principal d'activer et de désactiver les clés KMS, mais uniquement lorsque tous les alias des clés KMS incluent « `Test` ». Cette instruction de politique utilise deux conditions. La condition avec l'opérateur d'ensemble `ForAllValues` exige que tous les alias associés à la clé KMS incluent « Test ». La condition avec l'opérateur d'ensemble `ForAnyValue` exige que la clé KMS ait au moins un alias avec « Test ». Sans la condition `ForAnyValue`, cette instruction de politique aurait autorisé le principal à utiliser les clés KMS sans alias.

```
{
  "Sid": "AliasBasedIAMPolicy",
  "Effect": "Allow",
  "Action": [
    "kms:EnableKey",
    "kms:DisableKey"
  ],
  "Resource": "arn:aws:kms:*:111122223333:key/*",
  "Condition": {
    "ForAllValues:StringLike": {
      "kms:ResourceAliases": [
        "alias/*Test*"
      ]
    },
    "ForAnyValue:StringLike": {
      "kms:ResourceAliases": [
        "alias/*Test*"
      ]
    }
  }
}
```

km : ReplicaRegion

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ReplicaRegion`	Chaîne (liste)	À valeur unique	`ReplicateKey`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour limiter la Régions AWS capacité d'un principal à répliquer une clé multirégionale. La clé de `kms:ReplicaRegion` condition contrôle l'accès à l'ReplicateKeyopération en fonction de la valeur du ReplicaRegionparamètre dans la demande. Ce paramètre spécifie la Région AWS pour la nouvelle clé de réplica.

La valeur de la condition est un ou plusieurs Région AWS noms, tels que `us-east-1` ou`ap-southeast-2`, ou des modèles de noms, tels que`eu-*`. Pour obtenir la liste des noms de Régions AWS ces AWS KMS supports, consultez la section AWS Key Management Service Points de terminaison et quotas dans le Références générales AWS.

Par exemple, la déclaration de politique clé suivante utilise la clé de `kms:ReplicaRegion` condition pour permettre aux principaux d'appeler l'ReplicateKeyopération uniquement lorsque la valeur du `ReplicaRegion` paramètre est l'une des régions spécifiées.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Administrator"
  },
  "Action": "kms:ReplicateKey"
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ReplicaRegion": [
         "us-east-1",
         "eu-west-3",
         "ap-southeast-2"
      ]
    }
  }
}
```

Cette clé de condition contrôle uniquement l'accès à l'ReplicateKeyopération. Pour contrôler l'accès à l'UpdatePrimaryRegionopération, utilisez la clé de PrimaryRegion condition kms :.

km : RetiringPrincipal

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:RetiringPrincipal`	Chaîne (liste)	À valeur unique	`CreateGrant`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour contrôler l'accès à l'CreateGrantopération en fonction de la valeur du RetiringPrincipalparamètre dans la demande. Par exemple, vous pouvez créer des octrois d'utilisation d'une clé KMS uniquement lorsque le `RetiringPrincipal` de la demande `CreateGrant` correspond au `RetiringPrincipal` spécifié dans l'instruction de condition.

Pour spécifier le principal sortant, utilisez le Amazon Resource Name (ARN) d'un AWS principal. Les principaux valides incluent les utilisateurs IAM Comptes AWS, les rôles IAM, les utilisateurs fédérés et les utilisateurs des rôles assumés. Pour obtenir de l'aide sur la syntaxe ARN d'un principal, consultez IAM ARNs dans le guide de l'utilisateur IAM.

L'exemple de déclaration de politique clé suivant permet à un utilisateur de créer des autorisations pour la clé KMS. La clé de `kms:RetiringPrincipal` condition restreint l'autorisation aux `CreateGrant` demandes pour lesquelles le principal sortant de la subvention est le. `LimitedAdminRole`

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:RetiringPrincipal": "arn:aws:iam::111122223333:role/LimitedAdminRole"
    }
  }
}
```

Voir aussi

- km : GrantConstraintType
- km : GrantIsFor AWSResource
- km : GrantOperations
- km : GranteePrincipal

km : RotationPeriodInDays

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:RotationPeriodInDays`	Numérique	À valeur unique	`EnableKeyRotation`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour limiter les valeurs que les principaux peuvent spécifier dans le `RotationPeriodInDays` paramètre d'une EnableKeyRotationdemande.

`RotationPeriodInDays`Spécifie le nombre de jours entre chaque date de rotation automatique des clés. AWS KMS vous permet de spécifier une période de rotation comprise entre 90 et 2560 jours, mais vous pouvez utiliser la clé de `kms:RotationPeriodInDays` condition pour restreindre davantage la période de rotation, par exemple en imposant une période de rotation minimale dans la plage valide.

Par exemple, la déclaration de politique clé suivante utilise la clé de `kms:RotationPeriodInDays` condition pour empêcher les directeurs d'activer la rotation des clés si la période de rotation est inférieure ou égale à 180 jours.

```
{
  "Effect": "Deny",
  "Action": "kms:EnableKeyRotation",
  "Principal": "*",
  "Resource": "*",
  "Condition" : {
      "NumericLessThanEquals" : {
        "kms:RotationPeriodInDays" : "180"
      }
  }
```

```
}
```

km : ScheduleKeyDeletionPendingWindowInDays

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ScheduleKeyDeletionPendingWindowInDays`	Numérique	À valeur unique	`ScheduleKeyDeletion`	Politiques de clé et politiques IAM

Vous pouvez utiliser cette clé de condition pour limiter les valeurs que les principaux peuvent spécifier dans le `PendingWindowInDays` paramètre d'une ScheduleKeyDeletiondemande.

`PendingWindowInDays`Spécifie le nombre de jours à AWS KMS attendre avant de supprimer une clé. AWS KMS vous permet de spécifier une période d'attente comprise entre 7 et 30 jours, mais vous pouvez utiliser la clé de `kms:ScheduleKeyDeletionPendingWindowInDays` condition pour limiter davantage la période d'attente, par exemple en imposant une période d'attente minimale comprise dans la plage valide.

Par exemple, l'instruction de politique de clé suivante utilise la clé de condition `kms:ScheduleKeyDeletionPendingWindowInDays` pour empêcher les principaux de planifier la suppression des clés si le délai d'attente est inférieur ou égal à 21 jours.

```
{
  "Effect": "Deny",
  "Action": "kms:ScheduleKeyDeletion",
  "Principal": "*",
  "Resource": "*",
  "Condition" : {
      "NumericLessThanEquals" : {
        "kms:ScheduleKeyDeletionPendingWindowInDays" : "21"
      }
  }
}
```

km : SigningAlgorithm

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:SigningAlgorithm`	Chaîne	À valeur unique	`Sign` `Verify`	Politiques de clé et politiques IAM

Vous pouvez utiliser la clé de `kms:SigningAlgorithm` condition pour contrôler l'accès aux opérations de signature et de vérification en fonction de la valeur du SigningAlgorithmparamètre dans la demande. Cette clé de condition n'a aucun effet sur les opérations effectuées en dehors de AWS KMS, telles que la vérification des signatures avec la clé publique dans une paire de clés KMS asymétrique en dehors de AWS KMS.

L'exemple de politique de clé suivant permet aux utilisateurs pouvant endosser le rôle `testers` d'utiliser la clé KMS pour signer des messages uniquement lorsque l'algorithme de signature utilisé pour la demande est un algorithme RSASSA_PSS, tel que `RSASSA_PSS_SHA512`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/testers"
  },
  "Action": "kms:Sign",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:SigningAlgorithm": "RSASSA_PSS*"
    }
  }
}
```

Voir aussi

- km : EncryptionAlgorithm
- the section called "km : MacAlgorithm"
- the section called "km : MessageType"

km : ValidTo

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ValidTo`	Horodatage	À valeur unique	`ImportKeyMaterial`	Politiques de clé et politiques IAM

La clé de `kms:ValidTo` condition contrôle l'accès à l'ImportKeyMaterialopération en fonction de la valeur du ValidToparamètre dans la demande, qui détermine la date d'expiration du matériel clé importé. La valeur est exprimée en heure Unix.

Par défaut, le paramètre `ValidTo` est obligatoire dans une demande `ImportKeyMaterial`. Toutefois, si la valeur du ExpirationModelparamètre est`KEY_MATERIAL_DOES_NOT_EXPIRE`, le `ValidTo` paramètre n'est pas valide. Vous pouvez également utiliser la clé de ExpirationModel condition kms : pour demander le `ExpirationModel` paramètre ou une valeur de paramètre spécifique.

L'exemple d'instruction de politique suivant autorise un utilisateur à importer des éléments de clé dans une clé KMS. La clé de condition `kms:ValidTo` limite l'autorisation aux requêtes `ImportKeyMaterial` où la valeur `ValidTo` est inférieure ou égale à `1546257599.0` (31 décembre 2018 11:59:59 PM).

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:ImportKeyMaterial",
  "Resource": "*",
  "Condition": {
    "NumericLessThanEquals": {
      "kms:ValidTo": "1546257599.0"
    }
  }
}
```

Voir aussi

- km : ExpirationModel
- km : WrappingAlgorithm
- km : WrappingKeySpec

km : ViaService

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:ViaService`	Chaîne	À valeur unique	Opérations liées aux ressources de clé KMS	Politiques de clé et politiques IAM

La clé de `kms:ViaService` condition limite l'utilisation d'une clé KMS aux demandes provenant de AWS services spécifiques. Vous pouvez spécifier un ou plusieurs services dans chaque clé de condition `kms:ViaService`. L'opération doit être une opération de ressource de clé KMS, c'est-à-dire une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table Actions and Resources (Actions et ressources), recherchez la valeur de `KMS key` dans la colonne `Resources` de l'opération.

Par exemple, la déclaration de politique clé suivante utilise la clé de `kms:ViaService` condition pour autoriser l'utilisation d'une clé gérée par le client pour les actions spécifiées uniquement lorsque la demande provient d'Amazon EC2 ou d'Amazon RDS dans la région de l'ouest des États-Unis (Oregon) pour le compte de`ExampleRole`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:DescribeKey"
```

```
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "ec2.us-west-2.amazonaws.com",
        "rds.us-west-2.amazonaws.com"
      ]
    }
  }
}
```

Vous pouvez également utiliser une clé de condition `kms:ViaService` pour refuser l'autorisation d'utiliser une clé KMS lorsque la demande provient de services particuliers. Par exemple, l'instruction suivante de politique d'une politique de clé utilise une clé de condition `kms:ViaService` pour empêcher une clé gérée par le client d'être utilisée pour les opérations `Encrypt` lorsque la demande provient de AWS Lambda au nom de `ExampleRole`.

```
{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
          "lambda.us-west-2.amazonaws.com"
      ]
    }
  }
}
```

> **Important**
>
> Lorsque vous utilisez la clé de condition `kms:ViaService`, le service effectue la demande au nom d'un principal dans le Compte AWS. Ces principaux doivent disposer des autorisations suivantes :

- Autorisation d'utiliser la clé KMS. Le principal doit accorder ces autorisations au service intégré pour que celui-ci puisse utiliser la clé gérée par le client au nom du principal. Pour de plus amples informations, veuillez consulter Utilisation du AWS KMS chiffrement avec les AWS services.
- L'autorisation d'utiliser le service intégré. Pour en savoir plus sur l'accès des utilisateurs à un AWS service intégré AWS KMS, consultez la documentation du service intégré.

Toutes les Clés gérées par AWS utilisent une clé de condition `kms:ViaService` dans leur document de politique de clé. Cette condition permet à la clé KMS d'être utilisée uniquement pour les demandes qui proviennent du service qui a créé la clé KMS. Pour voir la politique clé d'un Clé gérée par AWS, utilisez l'GetKeyPolicyopération.

La clé de condition `kms:ViaService` est valide dans les instructions de la politique IAM et de la politique de clé. Les services que vous spécifiez doivent être intégrés à AWS KMS et prendre en charge la clé de condition `kms:ViaService`.

Services prenant en charge la clé de condition **`kms:ViaService`**

Le tableau suivant répertorie les AWS services intégrés AWS KMS et prenant en charge l'utilisation de la clé de `kms:ViaService` condition dans les clés gérées par le client. Les services de ce tableau peuvent ne pas être disponibles dans toutes les régions. Utilisez le `.amazonaws.com` suffixe du AWS KMS ViaService nom dans toutes les AWS partitions.

> **Note**
>
> Il peut être nécessaire de faire défiler horizontalement ou verticalement pour afficher toutes les données de ce tableau.

Nom du service	AWS KMS ViaService nom
Opération d'IA Amazon	`aiops.`*`AWS_region`*` .amazonaws.com`
AWS App Runner	`apprunner.`*`AWS_region`*`.amazonaws.com`

Nom du service	AWS KMS ViaService nom
AWS AppFabric	appfabric. *AWS_region* .amazonaws.com
Amazon AppFlow	appflow.*AWS_region* .amazonaws.com
AWS Application Migration Service	mgn.*AWS_region* .amazonaws.com
Amazon Athena	athena.*AWS_region* .amazonaws.com
AWS Audit Manager	auditmanager. *AWS_region* .amazonaws.com
Amazon Aurora	rds.*AWS_region* .amazonaws.com
AWS Backup	backup.*AWS_region* .amazonaws.com
Passerelle AWS Backup	backup-gateway. *AWS_region* .amazonaws.com
Copie du modèle Amazon Bedrock	bedrock.*AWS_region* .amazonaws.com
Kit SDK Amazon Chime	chimevoiceconnector. *AWS_region* .amazonaws.com
AWS Clean Rooms ML	cleanrooms-ml. *AWS_region* .amazonaws.com
AWS CodeArtifact	codeartifact. *AWS_region* .amazonaws.com
CodeGuru Réviseur Amazon	codeguru-reviewer. *AWS_region* .amazonaws.com
Amazon Comprehend	comprehend. *AWS_region* .amazonaws.com
Amazon Connect	connect.*AWS_region* .amazonaws.com

Nom du service	AWS KMS ViaService nom
Profils des clients Amazon Connect	`profile.AWS_region .amazonaws.com`
Amazon Q in Connect	`wisdom.AWS_region .amazonaws.com`
AWS Database Migration Service (AWS DMS)	`dms.AWS_region .amazonaws.com`
AWS DeepRacer	`deepracer. AWS_region .amazonaws.com`
AWS Directory Service	`directoryservice. AWS_region .amazonaws.com`
Amazon DocumentDB	`docdb-elastic. AWS_region .amazonaws.com`
Amazon DynamoDB	`dynamodb. AWS_region .amazonaws.com`
Amazon EC2 Systems Manager (SSM)	`ssm.AWS_region .amazonaws.com`
Amazon Elastic Block Store (Amazon EBS)	`ec2.AWS_region .amazonaws.com` (EBS uniquement)
Amazon Elastic Container Registry (Amazon ECR)	`ecr.AWS_region .amazonaws.com`
Amazon Elastic File System (Amazon EFS)	`elasticfilesystem. AWS_region .amazonaws.com`
Amazon ElastiCache	Incluez les deux ViaService noms dans la valeur de la clé de condition : • `elasticache. AWS_region .amazonaws.com` • `dax.AWS_region .amazonaws.com`
AWS Elemental MediaTailor	`mediatailor. AWS_region .amazonaws.com`

Nom du service	AWS KMS ViaService nom
AWS Résolution de l'entité	entityresolution. *AWS_region* .amazonaws.com
Amazon EventBridge	events.*AWS_region* .amazonaws.com
Amazon FinSpace	finspace. *AWS_region* .amazonaws.com
Amazon Forecast	forecast. *AWS_region* .amazonaws.com
Amazon FSx	fsx.*AWS_region* .amazonaws.com
AWS Glue	glue.*AWS_region* .amazonaws.com
AWS Ground Station	groundstation. *AWS_region* .amazonaws.com
Amazon GuardDuty	malware-protection. *AWS_region* .amazonaws.com
AWS HealthLake	healthlake. *AWS_region* .amazonaws.com
AWS IoT SiteWise	iotsitewise. *AWS_region* .amazonaws.com
Amazon Kendra	kendra.*AWS_region* .amazonaws.com
Amazon Keyspaces (pour Apache Cassandra)	cassandra. *AWS_region* .amazonaws.com
Amazon Kinesis	kinesis.*AWS_region* .amazonaws.com
Amazon Data Firehose	firehose. *AWS_region* .amazonaws.com

Nom du service	AWS KMS ViaService nom
Amazon Kinesis Video Streams	kinesisvideo. *AWS_region* .amazonaws.com
AWS Lambda	lambda.*AWS_region* .amazonaws.com
Amazon Lex	lex.*AWS_region* .amazonaws.com
AWS License Manager	license-manager. *AWS_region* .amazonaws.com
Amazon Location Service	geo.*AWS_region* .amazonaws.com
Amazon Lookout for Equipment	lookoutequipment. *AWS_region* .amazonaws.com
Amazon Lookout for Metrics	lookoutmetrics. *AWS_region* .amazonaws.com
Amazon Lookout for Vision	lookoutvision. *AWS_region* .amazonaws.com
Amazon Macie	macie.*AWS_region* .amazonaws.com
AWS Mainframe Modernization	m2.*AWS_region* .amazonaws.com
AWS Mainframe Modernization Tests d'applications	apptest.*AWS_region* .amazonaws.com
Amazon Managed Blockchain	managedblockchain. *AWS_region* .amazonaws.com
Amazon Managed Streaming for Apache Kafka (Amazon MSK)	kafka.*AWS_region* .amazonaws.com
Amazon Managed Workflows for Apache Airflow (MWAA)	airflow.*AWS_region* .amazonaws.com

Nom du service	AWS KMS ViaService nom
Amazon MemoryDB	memorydb. *AWS_region* .amazonaws.com
Amazon Monitron	monitron. *AWS_region* .amazonaws.com
Amazon MQ	mq.*AWS_region* .amazonaws.com
Amazon Neptune	rds.*AWS_region* .amazonaws.com
Amazon Nimble Studio	nimble.*AWS_region* .amazonaws.com
AWS HealthOmics	omics.*AWS_region* .amazonaws.com
Amazon OpenSearch Service	es.*AWS_region* .amazonaws.com , aoss.*AWS_region* .amazonaws.com
OpenSearch Forfaits personnalisés Amazon	custom-packages. *AWS_region* .amazonaws.com
AWS Proton	proton.*AWS_region* .amazonaws.com
Amazon Quantum Ledger Database (Amazon QLDB)	qldb.*AWS_region* .amazonaws.com
Analyse des performances d'Amazon RDS	rds.*AWS_region* .amazonaws.com
Amazon Redshift	redshift. *AWS_region* .amazonaws.com
Ouvrez l'éditeur de requête V2 Amazon Redshift.	sqlworkbench. *AWS_region* .amazonaws.com
Amazon Redshift Serverless	redshift-serverless. *AWS_region* .amazonaws.com
Amazon Rekognition	rekognition. *AWS_region* .amazonaws.com

Nom du service	AWS KMS ViaService nom
Amazon Relational Database Service (Amazon RDS)	rds.*AWS_region* .amazonaws.com
Stockage de données répliquées Amazon	ards.*AWS_region* .amazonaws.com
Amazon SageMaker AI	sagemaker. *AWS_region* .amazonaws.com
AWS Secrets Manager	secretsmanager. *AWS_region* .amazonaws.com
Amazon Security Lake	securitylake. *AWS_region* .amazonaws.com
Amazon Simple Email Service (Amazon SES)	ses.*AWS_region* .amazonaws.com
Amazon Simple Notification Service (Amazon SNS)	sns.*AWS_region* .amazonaws.com
Amazon Simple Queue Service (Amazon SQS)	sqs.*AWS_region* .amazonaws.com
Amazon Simple Storage Service (Amazon S3)	s3.*AWS_region* .amazonaws.com
Tables Amazon S3	s3tables. *AWS_region* .amazonaws.com
AWS Snowball Edge	importexport. *AWS_region* .amazonaws.com
AWS Step Functions	states.*AWS_region* .amazonaws.com
AWS Storage Gateway	storagegateway. *AWS_region* .amazonaws.com
AWS Systems Manager Incident Manager	ssm-incidents. *AWS_region* .amazonaws.com

Nom du service	AWS KMS ViaService nom
AWS Systems Manager Incident Manager Contacts	`ssm-contacts.` *`AWS_region`* `.amazonaws.com`
Amazon Timestream	`timestream.` *`AWS_region`* `.amazonaws.com`
Amazon Translate	`translate.` *`AWS_region`* `.amazonaws.com`
Accès vérifié par AWS	`verified-access.` *`AWS_region`* `.amazonaws.com`
Amazon WorkMail	`workmail.` *`AWS_region`* `.amazonaws.com`
Amazon WorkSpaces	`workspaces.` *`AWS_region`* `.amazonaws.com`
Amazon WorkSpaces Thin Client	`thinclient.` *`AWS_region`* `.amazonaws.com`
WorkSpaces Site Web d'Amazon	`workspaces-web.` *`AWS_region`* `.amazonaws.com`
AWS X-Ray	`xray.`*`AWS_region`* `.amazonaws.com`

km : WrappingAlgorithm

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:WrappingAlgorithm`	Chaîne	À valeur unique	`GetParametersForImport`	Politiques de clé et politiques IAM

Cette clé de condition contrôle l'accès à l'GetParametersForImportopération en fonction de la valeur du WrappingAlgorithmparamètre dans la demande. Vous pouvez utiliser cette condition pour exiger des principaux qu'ils utilisent un algorithme particulier pour chiffrer des matériaux clé au cours du processus d'importation. Les demandes de clé publique requise et du jeton d'importation échouent lorsqu'un algorithme d'encapsulage est spécifié.

L'exemple d'instruction de politique de clé suivant utilise la clé de condition `kms:WrappingAlgorithm` pour donner à l'utilisateur de l'exemple l'autorisation d'appeler l'opération `GetParametersForImport`, mais l'empêche d'utiliser l'algorithme d'encapsulage `RSAES_OAEP_SHA_1`. Lorsque `WrappingAlgorithm` dans la requête `GetParametersForImport` indique `RSAES_OAEP_SHA_1`, l'opération échoue.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:GetParametersForImport",
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "kms:WrappingAlgorithm": "RSAES_OAEP_SHA_1"
    }
  }
}
```

Voir aussi

- km : ExpirationModel
- km : ValidTo
- km : WrappingKeySpec

km : WrappingKeySpec

AWS KMS clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de stratégie
`kms:WrappingKeySpec`	Chaîne	À valeur unique	`GetParametersForImport`	Politiques de clé et politiques IAM

Cette clé de condition contrôle l'accès à l'GetParametersForImportopération en fonction de la valeur du WrappingKeySpecparamètre dans la demande. Vous pouvez utiliser cette condition pour exiger des principaux qu'ils utilisent un type particulier de clé publique au cours du processus d'importation. Si la requête spécifie un autre type de clé, elle échoue.

Étant donné que la seule valeur valide comme valeur du paramètre `WrappingKeySpec` est `RSA_2048`, les utilisateurs ne peuvent pas employer cette valeur de façon efficace, ce qui les empêche d'utiliser l'opération `GetParametersForImport`.

L'exemple d'instruction de politique suivant utilise la clé de condition `kms:WrappingAlgorithm` pour exiger que la valeur du paramètre `WrappingKeySpec` de la demande soit `RSA_4096`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:GetParametersForImport",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:WrappingKeySpec": "RSA_4096"
    }
  }
}
```

Voir aussi

- km : ExpirationModel
- km : ValidTo

- km : WrappingAlgorithm

AWS KMS clés de condition pour AWS Nitro Enclaves

AWS Nitro Enclaves est une EC2 fonctionnalité d'Amazon qui vous permet de créer des environnements informatiques isolés appelés enclaves pour protéger et traiter des données hautement sensibles. AWS KMS fournit des clés de condition pour prendre en charge AWS Nitro Enclaves. Ces clés de conditions ne sont efficaces que pour les demandes adressées à AWS KMS une Nitro Enclave.

Lorsque vous appelez les opérations Decrypt, DeriveSharedSecret, GenerateDataKeyGenerateDataKeyPair, ou GenerateRandomAPI avec le document d'attestation signé depuis une enclave, celles-ci APIs chiffrent le texte en clair de la réponse sous la clé publique du document d'attestation et renvoient du texte chiffré au lieu du texte en clair. Ce texte chiffré peut être déchiffré uniquement à l'aide de la clé privée dans l'enclave. Pour de plus amples informations, veuillez consulter Attestation cryptographique pour AWS Nitro Enclaves.

Les clés de condition suivantes vous permettent de limiter les autorisations pour ces opérations en fonction du contenu du document d'attestation signé. Avant d'autoriser une opération, AWS KMS compare le document d'attestation de l'enclave aux valeurs de ces clés de AWS KMS condition.

km RecipientAttestation : ImageSha 384

AWS KMS Clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de politique

`kms:RecipientAttestation:ImageSha384`	Chaîne	À valeur unique	`Decrypt`
			`DeriveSharedSecret`
			`GenerateDataKey`
			`GenerateDataKeyPair`
			`GenerateRandom`
				Politiques de clé et politiques IAM

La clé de `kms:RecipientAttestation:ImageSha384` condition contrôle l'accès à `DecryptDeriveSharedSecret`,`GenerateDataKey`,`GenerateDataKeyPair`, et `GenerateRandom` avec une clé KMS lorsque le résumé d'image du document d'attestation signé dans la demande correspond à la valeur de la clé de condition. La valeur `ImageSha384` correspond au PCR0 du document d'attestation. Cette clé de condition n'est effective que lorsque le `Recipient` paramètre de la demande spécifie un document d'attestation signé pour une enclave AWS Nitro.

Cette valeur est également incluse dans les CloudTrailévénements relatifs aux demandes adressées AWS KMS aux enclaves Nitro.

Par exemple, la déclaration de politique clé suivante autorise le `data-processing` rôle à utiliser la clé KMS pour les GenerateRandomopérations de déchiffrement DeriveSharedSecret, GenerateDataKey, GenerateDataKeyPair, et. La clé de condition `kms:RecipientAttestation:ImageSha384` permet les opérations uniquement lorsque la valeur de hachage de l'image (PCR0) du document d'attestation de la demande correspond à la valeur de hachage de l'image de la condition. Cette clé de condition n'est effective que lorsque le `Recipient` paramètre de la demande spécifie un document d'attestation signé pour une enclave AWS Nitro.

Si la demande n'inclut pas de document d'attestation valide provenant d'une enclave AWS Nitro, l'autorisation est refusée car cette condition n'est pas remplie.

```
{
  "Sid" : "Enable enclave data processing",
  "Effect" : "Allow",
  "Principal" : {
    "AWS" : "arn:aws:iam::111122223333:role/data-processing"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DeriveSharedSecret",
    "kms:GenerateDataKey",
    "kms:GenerateDataKeyPair",
    "kms:GenerateRandom"
  ],
  "Resource" : "*",
  "Condition": {
    "StringEqualsIgnoreCase": {
      "kms:RecipientAttestation:ImageSha384":
 "9fedcba8abcdef7abcdef6abcdef5abcdef4abcdef3abcdef2abcdef1abcdef1abcdef0abcdef1abcdef2abcdef3a
    }
  }
```

```
}
```

km : : PCR RecipientAttestation <PCR_ID>

AWS KMS Clés de condition	Type de condition	Type de la valeur	Opérations d'API	Type de politique

`kms:RecipientAttestation:PCR<PCR_ID>`	Chaîne	À valeur unique	`Decrypt`
			`DeriveSharedSecret`
			`GenerateDataKey`
			`GenerateDataKeyPair`
			`GenerateRandom`
				Politiques de clé et politiques IAM

La clé de `kms:RecipientAttestation:PCR<PCR_ID>` condition contrôle l'accès à`Decrypt`,`DeriveSharedSecret`, `GenerateDataKey``GenerateDataKeyPair`, et `GenerateRandom` avec une clé KMS uniquement lorsque les enregistrements de configuration de la plate-forme (PCRs) à partir du document d'attestation signé dans la demande correspondent PCRs à ceux de la clé de condition. Cette clé de condition n'est effective que lorsque le `Recipient` paramètre de la demande spécifie un document d'attestation signé provenant d'une enclave AWS Nitro.

Cette valeur est également incluse dans les CloudTrailévénements qui représentent des demandes adressées à AWS KMS des enclaves Nitro.

Pour spécifier une valeur PCR, utilisez le format suivant. Concaténez l'ID de PCR au nom de clé de condition. Vous pouvez spécifier un identifiant PCR identifiant l'une des six mesures de l'enclave ou un identifiant PCR personnalisé que vous avez défini pour un cas d'utilisation spécifique. La valeur PCR doit être une chaîne hexadécimale en minuscules de 96 octets maximum.

```
"kms:RecipientAttestation:PCRPCR_ID": "PCR_value"
```

Par exemple, la clé de condition suivante spécifie une valeur particulière pour PCR1, qui correspond au hachage du noyau utilisé pour l'enclave et le processus d'amorçage.

```
kms:RecipientAttestation:PCR1:
 "0x1abcdef2abcdef3abcdef4abcdef5abcdef6abcdef7abcdef8abcdef9abcdef8abcdef7abcdef6abcdef5abcdef
```

Par exemple, l'instruction de stratégie de clé suivante autorise le rôle `data-processing` à utiliser la clé KMS pour l'opération Decrypt.

La clé de `kms:RecipientAttestation:PCR` condition contenue dans cette instruction autorise l'opération uniquement lorsque la PCR1 valeur du document d'attestation signé dans la demande correspond à la `kms:RecipientAttestation:PCR1` valeur de la condition. Utilisez l'opérateur de politique `StringEqualsIgnoreCase` pour exiger une comparaison insensible à la casse des valeurs PCR.

Si la demande n'inclut pas de document d'attestation, l'autorisation est refusée car cette condition n'est pas remplie.

```
{
  "Sid" : "Enable enclave data processing",
  "Effect" : "Allow",
  "Principal" : {
    "AWS" : "arn:aws:iam::111122223333:role/data-processing"
  },
  "Action": "kms:Decrypt",
  "Resource" : "*",
  "Condition": {
    "StringEqualsIgnoreCase": {
      "kms:RecipientAttestation:PCR1":
 "0x1de4f2dcf774f6e3b679f62e5f120065b2e408dcea327bd1c9dddaea6664e7af79355814748447674530 82c6f15
    }
  }
}
```

Autorisations relatives au moindre privilège

Étant donné que vos clés KMS protègent les informations sensibles, nous vous recommandons de suivre le principe de l'accès le moins privilégié. Déléguez les autorisations minimales requises pour effectuer une tâche lorsque vous définissez vos politiques clés. N'autorisez toutes les actions (`kms:*`) sur une politique de clé KMS que si vous prévoyez de restreindre davantage les

autorisations avec des politiques IAM supplémentaires. Si vous envisagez de gérer les autorisations à l'aide de politiques IAM, limitez le nombre de personnes habilitées à créer et à associer des politiques IAM aux principaux IAM et surveillez les modifications apportées aux politiques.

Si vous autorisez toutes les actions (`kms:*`) à la fois dans la politique clé et dans la stratégie IAM, le principal dispose des autorisations d'administration et d'utilisation de la clé KMS. Pour des raisons de sécurité, nous recommandons de ne déléguer ces autorisations qu'à des responsables spécifiques. Vous pouvez le faire en nommant explicitement le principal dans la politique clé ou en limitant les principes auxquels la stratégie IAM est attachée. Vous pouvez également utiliser des clés de condition pour restreindre les autorisations. Par exemple, vous pouvez utiliser le `aws:PrincipalTag`pour autoriser toutes les actions si le principal effectuant l'appel d'API possède la balise spécifiée dans la règle de condition.

Pour mieux comprendre comment les déclarations de politique sont évaluées AWS, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM. Nous vous recommandons de consulter cette rubrique avant de rédiger des politiques afin de réduire le risque que votre politique ait des effets imprévus, tels que l'octroi d'un accès à des mandants qui ne devraient pas y avoir accès.

> **Tip**
>
> Lorsque vous testez une application dans un environnement hors production, utilisez IAM Access Analyzer pour vous aider à appliquer le principe du moindre privilège à vos politiques IAM.

Si vous utilisez des utilisateurs IAM plutôt que des rôles IAM, nous vous recommandons vivement d'activer l'authentification AWS multifactorielle (MFA) afin de réduire la vulnérabilité des informations d'identification à long terme. Vous pouvez utiliser MFA pour effectuer les tâches suivantes :

- Exigez que les utilisateurs valident leurs informations d'identification auprès de la MFA avant d'effectuer des actions privilégiées, telles que la planification de la suppression de clés.
- Répartissez la propriété du mot de passe d'un compte administrateur et du dispositif MFA entre les individus afin de mettre en œuvre une autorisation partagée.

En savoir plus

- AWS politiques gérées pour les fonctions professionnelles

- Techniques d'écriture de politiques IAM selon le principe de moindre privilège

Implémentation des autorisations avec le moindre privilégié

Lorsque vous autorisez un AWS service à utiliser une clé KMS, assurez-vous que l'autorisation n'est valide que pour les ressources auxquelles le service doit accéder en votre nom. Cette stratégie du moindre privilège permet d'empêcher l'utilisation non autorisée d'une clé KMS lorsque les demandes sont transmises entre les AWS services.

Pour mettre en œuvre une stratégie de moindre privilège, nous vous recommandons d'utiliser les clés de condition de contexte de AWS KMS chiffrement et les clés de condition de l'ARN source global ou du compte source.

Utilisation des clés de condition de contexte de chiffrement

Le moyen le plus efficace de mettre en œuvre les autorisations les moins privilégiées lors de l'utilisation AWS KMS des ressources consiste à inclure kms:EncryptionContext:touche contextuelle ou kms:EncryptionContextKeysclés de condition dans la politique qui permet aux principaux d'appeler des opérations AWS KMS cryptographiques. Ces clés de condition sont particulièrement efficaces parce qu'elles associent l'autorisation au contexte de chiffrement qui est lié au texte chiffré lorsque la ressource est chiffrée.

Utilisez les clés de conditions de contexte de chiffrement uniquement lorsque l'action indiquée dans la déclaration de politique est CreateGrantune opération cryptographique AWS KMS symétrique qui prend un `EncryptionContext` paramètre, telle que les opérations telles que GenerateDataKeyou Decrypt. (Pour une liste des opérations prises en charge, voir kms:EncryptionContext:touche contextuelle ou kms:EncryptionContextKeys.) Si vous utilisez ces clés de condition pour autoriser d'autres opérations, par exemple DescribeKey, l'autorisation sera refusée.

Définissez la valeur sur le contexte de chiffrement utilisé par le service lorsqu'il chiffre la ressource. Ces informations sont généralement disponibles dans le chapitre Sécurité de la documentation du service. Par exemple, le contexte de chiffrement de AWS Proton identifie la ressource AWS Proton et son modèle associé. Le contexte de chiffrement AWS Secrets Manager identifie le secret et sa version. Le contexte de chiffrement pour Amazon Location identifie le dispositif de suivi ou la collection.

L'exemple suivant d'instruction de politique de clé permet à Amazon Location Service de créer des octrois pour le compte des utilisateurs autorisés. Cette déclaration de politique limite l'autorisation

en utilisant les touches kms : CallerAccount, kms : et `kms:EncryptionContext:context-key` condition pour lier l'autorisation à une ressource de suivi particulière. ViaService

```
{
  "Sid": "Allow Amazon Location to create grants on behalf of authorized users",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/LocationTeam"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": "geo.us-west-2.amazonaws.com",
      "kms:CallerAccount": "111122223333",
      "kms:EncryptionContext:aws:geo:arn": "arn:aws:geo:us-west-2:111122223333:tracker/
SAMPLE-Tracker"
    }
  }
}
```

Utilisation des clés de condition **`aws:SourceArn`** ou **`aws:SourceAccount`**

Lorsque le principal indiqué dans une déclaration de politique clé est un directeur de AWS service, nous vous recommandons vivement d'utiliser le aws:SourceArn ou aws:SourceAccountclés de condition globales, en plus de la clé de `kms:EncryptionContext:context-key` condition. L'ARN et les valeurs du compte sont incluses dans le contexte d'autorisation uniquement lorsqu'une demande AWS KMS provient d'un autre AWS service. Cette combinaison de conditions implémente des autorisations de moindre privilège et évite l'éventualité pour un programme d'être manipulé par un autre pour obtenir un accès. Les principes de service ne sont généralement pas utilisés comme principes dans une politique clé, mais certains AWS services, tels que AWS CloudTrail, l'exigent.

Pour utiliser les clés de condition globales `aws:SourceArn` ou `aws:SourceAccount`, définissez comme valeur l'Amazon Resource Name (ARN) ou le compte de la ressource à chiffrer. Par exemple, dans une instruction de politique de clé qui autorise AWS CloudTrail à chiffrer un journal d'activité, définissez l'ARN de ce dernier comme valeur de `aws:SourceArn`. Dans la mesure du possible, utilisez `aws:SourceArn`, qui est plus spécifique. Définissez comme valeur l'ARN ou un modèle d'ARN avec des caractères génériques. Si vous ne connaissez pas l'ARN de la ressource, utilisez `aws:SourceAccount` à la place.

> **Note**
>
> Si un ARN de ressource inclut des caractères non autorisés dans une politique de AWS KMS clé, vous ne pouvez pas utiliser cet ARN de ressource dans la valeur de la clé de `aws:SourceArn` condition. Utilisez à la place la clé de condition `aws:SourceAccount`. Pour plus d'informations sur les règles de document de politique de clé, voir Format de politique de clé.

Dans l'exemple de politique de clé suivant, le principal qui obtient les autorisations est `cloudtrail.amazonaws.com`, le principal du service AWS CloudTrail . Pour implémenter le moindre privilège, cette politique utilise les clés de condition `aws:SourceArn` et `kms:EncryptionContext:`*`context-key`*. La déclaration de politique CloudTrail permet d'utiliser la clé KMS pour générer la clé de données utilisée pour chiffrer une trace. Les conditions `aws:SourceArn` et `kms:EncryptionContext:`*`context-key`* sont évaluées indépendamment. Toute demande d'utilisation de la clé KMS pour l'opération spécifiée doit répondre aux deux conditions.

Pour restreindre l'autorisation du service au journal d'activité `finance` dans l'exemple de compte (111122223333) et la région `us-west-2`, cette instruction de politique affecte à la condition de clé `aws:SourceArn` l'ARN d'un journal d'activité donné. L'instruction de condition utilise l'ArnEqualsopérateur pour garantir que chaque élément de l'ARN est évalué indépendamment lors de la correspondance. L'exemple utilise également la clé de condition `kms:EncryptionContext:`*`context-key`* pour limiter l'autorisation aux journaux d'activité dans un compte et une région particuliers.

Avant d'utiliser cette politique de clé, remplacez l'exemple d'ID de compte, de région et de nom de journal d'activité par des valeurs valides de votre compte.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow CloudTrail to encrypt logs",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "kms:GenerateDataKey",
```

```
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:cloudtrail:us-west-2:111122223333:trail/finance"
          ]
        },
        "StringLike": {
          "kms:EncryptionContext:aws:cloudtrail:arn": [
            "arn:aws:cloudtrail:*:111122223333:trail/*"
          ]
        }
      }
    }
  ]
}
```

ABAC pour AWS KMS

Le contrôle d'accès basé sur les attributs (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction des attributs. AWS KMS prend en charge ABAC en vous permettant de contrôler l'accès aux clés gérées par vos clients en fonction des balises et des alias associés aux clés KMS. Les clés de condition de balise et d'alias qui activent ABAC AWS KMS constituent un moyen puissant et flexible d'autoriser les principaux à utiliser les clés KMS sans modifier les politiques ni gérer les subventions. Toutefois, vous devriez utiliser cette fonction avec précaution, afin que les principaux ne soient pas autorisés ou refusés par inadvertance.

Si vous utilisez l'ABAC, sachez que l'autorisation de gérer les balises et les alias est désormais une autorisation de contrôle d'accès. Assurez-vous de connaître les balises et alias existants sur toutes les clés KMS avant de déployer une politique qui en dépend. Prenez des précautions raisonnables lors de l'ajout, de la suppression et de la mise à jour des alias, ainsi que lors de l'étiquetage et du désétiquetage des clés. Accordez des autorisations pour gérer les balises et les alias uniquement aux principaux qui en ont besoin, et limitez les balises et les alias qu'ils peuvent gérer.

> **Remarques**
>
> Lorsque vous utilisez ABAC pour AWS KMS, veillez à ne pas autoriser les principaux à gérer les balises et les alias. La modification d'une balise ou d'un alias peut autoriser ou refuser l'accès à une clé KMS. Les administrateurs de clés qui n'ont pas l'autorisation de modifier

les politiques de clé ou de créer des octrois peuvent contrôler l'accès aux clés KMS s'ils sont autorisés à gérer les balises ou les alias.
Les modifications d'alias et de balises peuvent prendre jusqu'à cinq minutes pour affecter l'autorisation de clé KMS. Les modifications récentes peuvent être visibles dans les opérations d'API avant qu'elles n'affectent l'autorisation.
Pour contrôler l'accès à une clé KMS en fonction de son alias, vous devez utiliser une clé de condition. Vous ne pouvez pas utiliser un alias pour représenter une clé KMS dans l'élément `Resource` d'une instruction de politique. Lorsqu'un alias apparaît dans l'élément `Resource`, l'instruction de politique s'applique à l'alias et non à la clé KMS associée.

En savoir plus

- Pour plus de détails sur la AWS KMS prise en charge d'ABAC, y compris des exemples, consultez Utiliser des alias pour contrôler l'accès aux clés KMS etUtiliser des balises pour contrôler l'accès aux clés KMS.
- Pour des informations plus générales sur l'utilisation de balises pour contrôler l'accès aux AWS ressources, voir À quoi sert ABAC ? AWS et le contrôle de l'accès aux AWS ressources à l'aide de balises de ressources dans le guide de l'utilisateur IAM.

Clés de condition ABAC pour AWS KMS

Pour autoriser l'accès aux clés KMS en fonction de leurs balises et alias, utilisez les clés de condition suivantes dans une politique de clé ou une politique IAM.

Clé de condition ABAC	Description	Type de stratégie	AWS KMS opérations
lois : ResourceTag	La balise (clé et valeur) de la clé KMS correspond à la balise (clé et valeur) ou au modèle de balise dans la politique.	Politique IAM uniquement	Opérations liées aux ressources de clé KMS [2]
aws :RequestTag//tag-key	La balise (clé et valeur) dans la	Politique de clé et politiques IAM[1]	TagResource, UntagResource

Clé de condition ABAC	Description	Type de stratégie	AWS KMS opérations
	demande correspond à la balise (clé et valeur) ou au modèle de balise dans la politique.		
lois : TagKeys	Dans la demande, les clés de balise correspondent à celles de la politique.	Politique de clé et politiques IAM[1]	TagResource, UntagResource
km : ResourceAliases	Les alias associés à la clé KMS correspondent aux alias ou aux modèles d'alias de la politique.	Politique IAM uniquement	Opérations liées aux ressources de clé KMS [2]
km : RequestAlias	L'alias qui représente la clé KMS dans la demande correspond à l'alias ou aux modèles d'alias de la politique.	Politique de clé et politiques IAM[1]	opérations cryptographiques, DescribeKeyGetPublicKey

[1]Toute clé de condition pouvant être utilisée dans une politique de clé peut également être utilisée dans une politique IAM, mais uniquement si la politique clé le permet.

[2]Une opération de ressource de clé KMS est une opération autorisée pour une clé KMS particulière. Pour identifier les opérations de ressources de clés KMS, dans la table AWS KMS des autorisations, recherchez la valeur de la clé KMS dans la colonne `Resources` de l'opération.

Par exemple, vous pouvez utiliser ces clés de condition pour créer les politiques suivantes.

- Une politique IAM avec `kms:ResourceAliases` qui autorise l'utilisation de clés KMS avec un alias ou un modèle d'alias particulier. Cela est un peu différent des politiques qui reposent sur des

balises : bien que vous puissiez utiliser des modèles d'alias dans une politique, chaque alias doit être unique dans une région Compte AWS et. Cela vous permet d'appliquer une politique à un ensemble sélectionné de clés KMS sans indiquer la clé ARNs des clés KMS dans la déclaration de stratégie. Pour ajouter ou supprimer des clés KMS de l'ensemble, modifiez l'alias de la clé KMS.

- Une politique de clé avec `kms:RequestAlias` qui permet aux principaux d'utiliser une clé KMS dans une opération `Encrypt`, mais uniquement lorsque la demande `Encrypt` utilise cet alias pour identifier la clé KMS.
- Une politique IAM avec `aws:ResourceTag/`*`tag-key`* qui refuse l'autorisation d'utiliser des clés KMS avec une clé et une valeur de balise particulières. Cela vous permet d'appliquer une politique à un ensemble sélectionné de clés KMS sans indiquer la clé ARNs des clés KMS dans la déclaration de stratégie. Pour ajouter ou supprimer des clés KMS de l'ensemble, étiquetez ou désétiquetez la clé KMS.
- Une politique IAM avec `aws:RequestTag/`*`tag-key`* qui permet aux principaux de supprimer uniquement les balises de clés KMS `"Purpose"="Test"`.
- Une politique IAM avec `aws:TagKeys` qui refuse l'autorisation d'étiqueter ou de désétiqueter une clé KMS avec une clé de balise `Restricted`.

L'ABAC rend la gestion des accès flexible et évolutive. Par exemple, vous pouvez utiliser la clé de condition `aws:ResourceTag/`*`tag-key`* pour créer une politique IAM qui permet aux principaux d'utiliser une clé KMS pour des opérations spécifiées uniquement lorsque la clé KMS possède une balise `Purpose=Test`. La politique s'applique à toutes les clés KMS dans toutes les régions du Compte AWS.

Lorsqu'elle est attachée à un utilisateur ou à un rôle, la politique IAM suivante permet aux principaux d'utiliser toutes les clés KMS existantes avec une balise `Purpose=Test` pour les opérations spécifiées. Pour autoriser cet accès à des clés KMS nouvelles ou existantes, vous n'avez pas besoin de modifier la politique. Il suffit de joindre la balise `Purpose=Test` aux clés KMS. De même, pour supprimer cet accès des clés KMS avec une balise `Purpose=Test`, modifiez ou supprimez la balise.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AliasBasedIAMPolicy",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
```

```
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Purpose": "Test"
        }
      }
    }
  ]
}
```

Toutefois, si vous utilisez cette fonction, faites attention lors de la gestion des balises et des alias. L'ajout, la modification ou la suppression d'une balise ou d'un alias peut autoriser ou refuser l'accès à une clé KMS par inadvertance. Les administrateurs de clés qui n'ont pas l'autorisation de modifier les politiques de clé ou de créer des octrois peuvent contrôler l'accès aux clés KMS s'ils sont autorisés à gérer les balises et les alias. Pour atténuer ce risque, envisagez de limiter les autorisations de gestion des balises et des alias. Par exemple, vous pouvez autoriser uniquement les principaux sélectionnés à gérer les balises `Purpose=Test`. Pour plus de détails, veuillez consulter Utiliser des alias pour contrôler l'accès aux clés KMS et Utiliser des balises pour contrôler l'accès aux clés KMS.

Des balises ou des alias ?

AWS KMS supporte ABAC avec des tags et des alias. Les deux options offrent une stratégie de contrôle d'accès flexible et évolutive, mais elles sont légèrement différentes l'une de l'autre.

Vous pouvez décider d'utiliser des balises ou des alias en fonction de vos habitudes AWS d'utilisation particulières. Par exemple, si vous avez déjà accordé des autorisations d'étiquetage à la plupart des administrateurs, il peut être plus facile de contrôler une stratégie d'autorisation basée sur des alias. Ou, si vous approchez du quota pour le nombre d'alias par clé KMS, vous pouvez préférer une stratégie d'autorisation basée sur des balises.

Les avantages suivants sont d'intérêt général.

Avantages du contrôle d'accès basé sur les identifications

- Même mécanisme d'autorisation pour différents types de AWS ressources.

Vous pouvez utiliser la même balise ou clé de balise pour contrôler l'accès à plusieurs types de ressources, tels qu'un cluster Amazon Relational Database Service (Amazon RDS), un volume Amazon Elastic Block Store (Amazon EBS) et une clé KMS. Cette fonction permet plusieurs modèles d'autorisation plus flexibles que le contrôle d'accès classique basé sur les rôles.

- Autoriser l'accès à un groupe de clés KMS.

  Vous pouvez utiliser des balises pour gérer l'accès à un groupe de clés KMS dans le même Compte AWS et la même région. Attribuez la même balise ou la même clé de balise aux clés KMS que vous choisissez. Créez ensuite une déclaration de easy-to-maintain politique simple basée sur le tag ou la clé du tag. Pour ajouter ou supprimer une clé KMS de votre groupe d'autorisations, ajoutez ou supprimez la balise ; vous n'avez pas besoin de modifier la politique.

Avantages du contrôle d'accès basé sur les alias

- Autoriser l'accès aux opérations cryptographiques en fonction des alias.

  La plupart des conditions de politique basées sur les demandes pour les attributs, y compris aws :RequestTag/tag-key, affectent uniquement les opérations qui ajoutent, modifient ou suppriment l'attribut. Mais la clé de RequestAlias condition kms : contrôle l'accès aux opérations cryptographiques en fonction de l'alias utilisé pour identifier la clé KMS dans la demande. Par exemple, vous pouvez accorder à un principal l'autorisation d'utiliser une clé KMS dans une opération `Encrypt` mais uniquement lorsque la valeur du paramètre `KeyId` est `alias/restricted-key-1`. Cette condition nécessite tous les éléments suivants pour répondre aux exigences :

  - La clé KMS doit être associée à cet alias.
  - La demande doit utiliser l'alias pour identifier la clé KMS.
  - Le principal doit être autorisé à utiliser la clé KMS sujette à la condition `kms:RequestAlias`.

  Cela est particulièrement utile si vos applications utilisent fréquemment des noms d'alias ou des alias ARNs pour faire référence aux clés KMS.

- Fournir des autorisations très limitées.

  Un alias doit être unique dans une région Compte AWS et. Par conséquent, donner aux principaux accès à une clé KMS basée sur un alias peut être beaucoup plus restrictif que leur donner un accès basé sur une balise. Contrairement aux alias, les balises peuvent être affectées à plusieurs clés KMS dans le même compte et la même région. Si vous le souhaitez, vous pouvez utiliser un

modèle d'alias, tel que `alias/test*`, pour donner aux principaux accès à un groupe de clés KMS dans le même compte et la même région. Cependant, autoriser ou refuser l'accès à un alias particulier permet un contrôle très strict sur les clés KMS.

Résolution des problèmes liés à ABAC pour AWS KMS

Le contrôle de l'accès aux clés KMS en fonction de leurs balises et alias est pratique et puissant. Cependant, cette méthode est sujette à quelques erreurs prévisibles que vous voudrez éviter.

Accès modifié en raison d'un changement de balise

Si une balise est supprimée ou si sa valeur est modifiée, les principaux qui ont accès à une clé KMS basée uniquement sur cette balise se verront refuser l'accès à la clé KMS. Cela peut également se produire lorsqu'une balise incluse dans une instruction de politique de refus est ajoutée à une clé KMS. L'ajout d'une balise liée à une politique à une clé KMS peut permettre l'accès aux principaux qui devraient se voir refuser l'accès à une clé KMS.

Supposons, par exemple, qu'un principal ait accès à une clé KMS basée sur la balise `Project=Alpha`, par exemple l'autorisation fournie par l'exemple d'instruction de politique IAM suivant.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:ap-southeast-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

Si la balise est supprimée de cette clé KMS ou si la valeur de la balise est modifiée, le principal n'a plus l'autorisation d'utiliser la clé KMS pour les opérations spécifiées. Cela peut devenir évident lorsque le directeur essaie de lire ou d'écrire des données dans un AWS service qui utilise une clé gérée par le client. Pour suivre le changement de balise, consultez vos CloudTrail journaux TagResourceou UntagResource entrées.

Pour restaurer l'accès sans mettre à jour la politique, modifiez les balises de la clé KMS. Cette mesure a un impact minime sur une brève période et elle prend effet sur l'ensemble de AWS KMS. Pour éviter une erreur comme celle-ci, accordez des autorisations d'étiquetage et de désétiquetage uniquement aux principaux qui en ont besoin et limitez leurs autorisations d'étiquetage aux balises qu'ils doivent gérer. Avant de modifier une balise, recherchez des politiques pour détecter l'accès qui dépend de la balise et obtenir des clés KMS dans toutes les régions qui possèdent la balise. Vous pouvez envisager de créer une CloudWatch alarme Amazon lorsque des balises spécifiques sont modifiées.

Changement d'accès dû à un changement d'alias

Si un alias est supprimé ou associé à une autre clé KMS, les principaux qui ont accès à la clé KMS basée uniquement sur cet alias se verront refuser l'accès à la clé KMS. Cela peut également se produire lorsqu'un alias associé à une clé KMS est inclus dans une instruction de politique de refus. L'ajout d'un alias lié à une politique à une clé KMS peut également permettre l'accès aux principaux qui devraient se voir refuser l'accès à une clé KMS.

Par exemple, la déclaration de politique IAM suivante utilise la clé de ResourceAliases condition kms : pour autoriser l'accès aux clés KMS dans les différentes régions du compte avec l'un des alias spécifiés.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AliasBasedIAMPolicy",
      "Effect": "Allow",
      "Action": [
        "kms:List*",
        "kms:Describe*",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
```

```
        "ForAnyValue:StringEquals": {
          "kms:ResourceAliases": [
            "alias/ProjectAlpha",
            "alias/ProjectAlpha_Test",
            "alias/ProjectAlpha_Dev"
          ]
        }
      }
    }
  ]
}
```

Pour suivre le changement d'alias, consultez vos CloudTrail journaux pour CreateAliasUpdateAlias, et vos DeleteAliasentrées.

Pour restaurer l'accès sans mettre à jour la politique, modifiez les alias associés à la clé KMS. Étant donné que chaque alias ne peut être associé qu'à une seule clé KMS dans un compte et une région, la gestion des alias est un peu plus difficile que la gestion des balises. La restauration de l'accès de certains principaux sur une clé KMS peut refuser au même ou à d'autres principaux l'accès à une autre clé KMS.

Pour éviter cette erreur, n'accordez des autorisations de gestion d'alias qu'aux principaux qui en ont besoin et limitez leurs autorisations de gestion des alias aux alias qu'ils doivent gérer. Avant de mettre à jour ou de supprimer un alias, recherchez des politiques pour détecter l'accès qui dépend de l'alias et recherchez les clés KMS dans toutes les régions associées à l'alias.

Accès refusé en raison d'un quota d'alias

Les utilisateurs autorisés à utiliser une clé KMS dans une limite de kilomètres bénéficieront ResourceAliases d'une `AccessDenied` exception si la clé KMS dépasse les alias par défaut par quota de clé KMS pour ce compte et cette région.

Pour restaurer l'accès, supprimez les alias associés à la clé KMS afin qu'elle soit conforme au quota. Sinon, utilisez un autre mécanisme pour accorder aux utilisateurs l'accès à la clé KMS.

Modification retardée de l'autorisation

Les modifications que vous apportez aux balises et aux alias peuvent prendre jusqu'à cinq minutes pour affecter l'autorisation des clés KMS. Par conséquent, un changement de balise ou d'alias peut être reflété dans les réponses des opérations d'API avant qu'elles n'affectent l'autorisation. Ce délai

est susceptible d'être plus long que le bref délai de cohérence éventuel qui affecte la plupart des AWS KMS opérations.

Par exemple, vous disposez peut-être d'une politique IAM qui autorise certains principaux à utiliser n'importe quelle clé KMS avec une balise `"Purpose"="Test"`. Ensuite, vous ajoutez la balise `"Purpose"="Test"` sur une clé KMS. Bien que l'TagResourceopération soit terminée et que la ListResourceTagsréponse confirme que la balise est attribuée à la clé KMS, les principaux peuvent ne pas avoir accès à la clé KMS pendant cinq minutes au maximum.

Pour éviter les erreurs, intégrez ce délai attendu à votre code.

Demandes ayant échoué en raison des mises à jour d'alias

Lorsque vous mettez à jour un alias, vous associez un alias existant à une autre clé KMS.

Le déchiffrement et les ReEncryptdemandes spécifiant le nom d'alias ou l'ARN de l'alias peuvent échouer car l'alias est désormais associé à une clé KMS qui n'a pas chiffré le texte chiffré. Cette situation renvoie généralement un `IncorrectKeyException` ou `NotFoundException`. Si la demande n'a pas de paramètre `KeyId` ou `DestinationKeyId`, l'opération peut échouer avec l'exception `AccessDenied`, car l'appelant n'a plus accès à la clé KMS qui a chiffré le texte chiffré.

Vous pouvez suivre les modifications en consultant CloudTrail les journaux et CreateAliasUpdateAliasles entrées des DeleteAliasjournaux. Vous pouvez également utiliser la valeur du `LastUpdatedDate` champ dans la ListAliasesréponse pour détecter un changement.

Par exemple, l'ListAliasesexemple de réponse suivant montre que l'`ProjectAlpha_Test`alias de la `kms:ResourceAliases` condition a été mis à jour. Par conséquent, les principaux qui ont un accès en fonction de l'alias perdent leur accès à la clé KMS précédemment associée. Au lieu de cela, ils ont accès à la clé KMS nouvellement associée.

```
$ aws kms list-aliases --query 'Aliases[?starts_with(AliasName, `alias/ProjectAlpha`)]'

{
    "Aliases": [
        {
            "AliasName": "alias/ProjectAlpha_Test",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ProjectAlpha_Test",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "CreationDate": 1566518783.394,
            "LastUpdatedDate": 1605308931.903
```

```
        },
        {
            "AliasName": "alias/ProjectAlpha_Restricted",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/
ProjectAlpha_Restricted",
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1553410800.010,
            "LastUpdatedDate": 1553410800.010
        }
    ]
}
```

La solution à ce problème n'est pas simple. Vous pouvez à nouveau mettre à jour l'alias pour l'associer à la clé KMS d'origine. Toutefois, avant d'agir, vous devez tenir compte de l'effet de cette modification sur la clé KMS actuellement associée. Si les principaux utilisent cette dernière clé KMS dans des opérations de chiffrement, ils peuvent avoir besoin d'un accès continu à celle-ci. Dans ce cas, vous pouvez mettre à jour la politique pour vous assurer que les principaux ont l'autorisation d'utiliser les deux clés KMS.

Vous pouvez empêcher une erreur comme celle-ci : avant de mettre à jour un alias, examinez les politiques pour détecter l'accès qui dépend de l'alias. Obtenez ensuite les clés KMS dans toutes les régions associées à l'alias. Accordez des autorisations de gestion d'alias uniquement aux principaux qui en ont besoin et limitez leurs autorisations de gestion d'alias aux alias qu'ils doivent gérer.

RBAC pour AWS KMS

Le contrôle d'accès basé sur les rôles (RBAC) est une stratégie d'autorisation qui fournit aux utilisateurs uniquement les autorisations nécessaires pour effectuer leurs tâches, et rien de plus. AWS KMS prend en charge le RBAC en vous permettant de contrôler l'accès à vos clés en spécifiant des autorisations détaillées sur l'utilisation des clés dans le cadre des politiques clés. Les politiques clés spécifient une ressource, une action, un effet, un principal et des conditions facultatives pour accorder l'accès aux clés.

Pour implémenter le RBAC dans AWS KMS, nous recommandons de séparer les autorisations pour les utilisateurs clés et les administrateurs principaux.

Key users

L'exemple de politique clé suivant permet au rôle `ExampleUserRole` IAM d'utiliser la clé KMS.

```
{
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws: iam::111122223333:role/ExampleUserRole"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
  }
```

Vos utilisateurs principaux peuvent avoir besoin de moins d'autorisations que l'utilisateur dans cet exemple. Attribuez uniquement les autorisations dont l'utilisateur a besoin. Posez les questions suivantes pour affiner davantage les autorisations.

- Quels sont les principaux IAM (rôles ou utilisateurs) qui ont besoin d'accéder à la clé ?
- Quelles actions chaque directeur doit-il effectuer avec la clé ? Par exemple, le principal a-t-il uniquement besoin des autorisations de chiffrement et de signature ?
- L'utilisateur est-il un être humain ou un AWS service ? S'il s'agit d'un AWS service, vous pouvez utiliser la clé de condition pour limiter l'utilisation des clés à un AWS service spécifique.

Key administrators

L'exemple de politique clé suivant permet au rôle `ExampleAdminRole` IAM d'administrer la clé KMS.

```
{
            "Sid": "Allow access for Key Administrators",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws: iam::111122223333:role/ExampleAdminRole"
            },
            "Action": [
                "kms:Create*",
```

```
                "kms:Describe*",
                "kms:Enable*",
                "kms:List*",
                "kms:Put*",
                "kms:Update*",
                "kms:Revoke*",
                "kms:Disable*",
                "kms:Get*",
                "kms:Delete*",
                "kms:TagResource",
                "kms:UntagResource",
                "kms:ScheduleKeyDeletion",
                "kms:CancelKeyDeletion"
            ],
            "Resource": "*"
    }
```

Vos administrateurs principaux peuvent avoir besoin de moins d'autorisations que l'administrateur dans cet exemple. N'attribuez que les autorisations dont vos principaux administrateurs ont besoin.

N'accordez aux utilisateurs que les autorisations dont ils ont besoin pour remplir leurs rôles. Les autorisations d'un utilisateur peuvent varier selon que la clé est utilisée dans des environnements de test ou de production. Si vous utilisez des autorisations moins restrictives dans certains environnements non liés à la production, mettez en œuvre un processus pour tester les politiques avant leur mise en production.

En savoir plus

- Identités IAM (utilisateurs, groupes d'utilisateurs et rôles)
- Types de contrôle d'accès

Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS

Vous pouvez autoriser les utilisateurs ou les rôles d'un autre Compte AWS pays à utiliser une clé KMS dans votre compte. L'accès inter-comptes nécessite une autorisation dans la politique de clé de la clé KMS et dans une politique IAM dans le compte de l'utilisateur externe.

L'autorisation inter-comptes n'est effective que pour les opérations suivantes :

- Opérations cryptographiques
- CreateGrant
- DescribeKey
- GetKeyRotationStatus
- GetPublicKey
- ListGrants
- RetireGrant
- RevokeGrant

Si vous accordez à un utilisateur d'un autre compte des autorisations pour d'autres opérations, ces autorisations n'ont aucun effet. Par exemple, si vous accordez au principal d'un autre compte une ListKeys autorisation kms : dans une politique IAM, ou kms : une ScheduleKeyDeletion autorisation sur une clé KMS dans une politique clé, les tentatives de l'utilisateur pour appeler ces opérations sur vos ressources échouent toujours.

Pour plus de détails sur l'utilisation des clés KMS dans différents comptes pour les AWS KMS opérations, consultez la colonne Utilisation entre comptes dans le AWS KMS autorisations etUtilisation de clés KMS dans d'autres comptes. Il existe aussi une section Cross-account use (Utilisation inter-comptes) dans chaque description d'API de la référence d'API AWS Key Management Service.

> **Warning**
>
> Soyez prudent lorsque vous autorisez les principaux à utiliser vos clés KMS. Dans la mesure du possible, suivez le principe du moindre privilège. Donnez uniquement aux utilisateurs l'accès aux clés KMS dont ils ont besoin pour les opérations dont ils ont besoin.
> Par ailleurs, soyez prudent en ce qui concerne l'utilisation d'une clé KMS inconnue, en particulier d'une clé KMS dans un compte différent. Les utilisateurs malveillants peuvent vous autoriser à utiliser leur clé KMS pour obtenir des informations sur vous ou votre compte.
> Pour plus d'informations sur l'utilisation des politiques pour protéger les ressources de votre compte, veuillez consulter Bonnes pratiques pour les politiques IAM.

Pour accorder l'autorisation d'utiliser une clé KMS aux utilisateurs et aux rôles d'un autre compte, vous devez utiliser deux types de politiques différents :

- La politique de clé pour la clé KMS doit accorder au compte externe (ou aux utilisateurs et rôles du compte externe) l'autorisation d'utiliser la clé KMS. La politique de clé se trouve dans le compte qui possède la clé KMS.
- Les politiques IAM du compte externe doivent déléguer les autorisations de politique de clé à leurs utilisateurs et rôles. Ces politiques sont définies dans le compte externe et accordent des autorisations aux utilisateurs et rôles de ce compte.

La politique de clé détermine qui peut avoir accès à la clé KMS. La politique IAM détermine qui a accès à la clé KMS. Ni la politique de clé ni la politique IAM à elles seules ne suffisent. Vous devez modifier les deux.

Pour modifier la politique clé, vous pouvez utiliser la vue des politiques dans les opérations AWS Management Console ou utiliser les PutKeyPolicyopérations CreateKeyou.

Pour obtenir de l'aide concernant la modification des politiques IAM, veuillez consulter Utilisation des politiques IAM avec AWS KMS.

Pour obtenir un exemple qui montre comment la politique de clé et les politiques IAM fonctionnent ensemble pour autoriser l'utilisation d'une clé KMS dans un autre compte, veuillez consulter Exemple 2 : L'utilisateur assume un rôle autorisé à utiliser une clé KMS dans un autre Compte AWS.

Vous pouvez consulter les AWS KMS opérations entre comptes qui en résultent sur la clé KMS dans vos AWS CloudTrail journaux. Les opérations qui utilisent des clés KMS dans d'autres comptes sont journalisées à la fois dans le compte de l'appelant et dans le compte propriétaire de la clé KMS.

Rubriques

- Étape 1 : ajouter une déclaration de politique de clé dans le compte local
- Étape 2 : ajouter des politiques IAM dans le compte externe
- Autoriser l'utilisation de clés KMS externes avec Services AWS
- Utilisation de clés KMS dans d'autres comptes

> **Note**
>
> Les exemples de cette rubrique montrent comment utiliser ensemble une politique de clé et une politique IAM pour fournir et limiter l'accès à une clé KMS. Ces exemples génériques ne sont pas destinés à représenter les autorisations requises par une clé KMS en particulier

Service AWS . Pour plus d'informations sur les autorisations requises par AN Service AWS , consultez la rubrique relative au chiffrement dans la documentation du service.

Étape 1 : ajouter une déclaration de politique de clé dans le compte local

La politique de clé pour une clé KMS constitue l'élément principal qui détermine qui peut accéder à la clé KMS et quelles sont les opérations pouvant être effectuées. La politique de clé est toujours définie dans le compte propriétaire de la clé KMS. Contrairement aux politiques IAM, les politiques de clé ne spécifient pas de ressource. La ressource est la clé KMS associée à la politique de clé. Lors de l'octroi d'une autorisation entre comptes, la politique de clé relative à la clé KMS doit accorder au compte externe (ou aux utilisateurs et rôles du compte externe) l'autorisation d'utiliser la clé KMS.

Pour accorder à un compte externe l'autorisation d'utiliser la clé KMS, ajoutez une instruction à la politique de clé qui spécifie le compte externe. Dans l'élément `Principal` de la politique de clé, entrez l'Amazon Resource Name (ARN) du compte externe.

Lorsque vous spécifiez un compte externe dans une politique de clé, les administrateurs IAM du compte externe peuvent utiliser des politiques IAM pour déléguer ces autorisations à tous les utilisateurs et rôles du compte externe. Ils peuvent également décider quelles sont les actions spécifiées dans la politique de clé que les utilisateurs et les rôles peuvent effectuer.

Les autorisations accordées au compte externe et à ses principaux ne sont efficaces que si le compte externe est activé dans la région qui héberge la clé KMS et sa politique de clé. Pour plus d'informations sur les régions qui ne sont pas activées par défaut (« Régions d'adhésion »), veuillez consulter Gestion de Régions AWS dans la Références générales AWS.

Par exemple, supposons que vous vouliez autoriser le compte `444455556666` à utiliser une clé KMS de chiffrement symétrique dans le compte `111122223333`. Pour ce faire, ajoutez une instruction de politique comme celle de l'exemple suivant à la politique de clé pour la clé KMS dans le compte `111122223333`. Cette instruction de politique accorde au compte externe, `444455556666`, l'autorisation d'utiliser la clé KMS dans les opérations de chiffrement pour les clés KMS de chiffrement symétriques.

> **Note**
>
> L'exemple suivant illustre une politique de clé qui permet de partager une clé KMS avec un autre compte. Remplacez les valeurs `Sid`, `Principal` et `Action` de l'exemple par des valeurs valides pour l'utilisation prévue de votre clé KMS.

```
{
    "Sid": "Allow an external account to use this KMS key",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::444455556666:root"
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
}
```

Au lieu d'accorder l'autorisation au compte externe, vous pouvez spécifier des utilisateurs et des rôles externes spécifiques dans la politique de clé. Toutefois, ces utilisateurs et rôles ne peuvent pas utiliser la clé KMS tant que les administrateurs IAM du compte externe n'ont pas attaché les politiques IAM appropriées à leurs identités. Les politiques IAM peuvent accorder une autorisation à tous les utilisateurs et rôles externes, ou à une partie d'entre eux seulement, qui sont spécifiés dans la politique de clé. Elles peuvent également autoriser tout ou partie des actions spécifiées dans la politique de clé.

La spécification d'identités dans une politique de clé restreint les autorisations que les administrateurs IAM du compte externe peuvent fournir. Toutefois, cela rend la gestion des politiques avec deux comptes plus complexe. Par exemple, supposons que vous ayez besoin d'ajouter un utilisateur ou un rôle. Vous devez ajouter cette identité à la politique de clé dans le compte propriétaire de la clé KMS et créer des politiques IAM dans le compte de l'identité.

Pour spécifier des utilisateurs ou des rôles externes spécifiques dans une politique de clé, dans l'élément `Principal`, entrez l'Amazon Resource Name (ARN) d'un utilisateur ou d'un rôle dans le compte externe.

Ainsi, l'exemple d'instruction de politique de clé suivant autorise le rôle `ExampleRole` du compte `444455556666` à utiliser une clé KMS du compte `111122223333`. Cette instruction de politique de clé accorde au compte externe, `444455556666`, l'autorisation d'utiliser la clé KMS dans les opérations de chiffrement pour les clés KMS de chiffrement symétriques.

> **Note**
>
> L'exemple suivant illustre une politique de clé qui permet de partager une clé KMS avec un autre compte. Remplacez les valeurs `Sid`, `Principal` et `Action` de l'exemple par des valeurs valides pour l'utilisation prévue de votre clé KMS.

```
{
    "Sid": "Allow an external account to use this KMS key",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::444455556666:role/ExampleRole"
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
}
```

> **Note**
>
> Ne définissez pas le principal sur un astérisque (*) dans une instruction de politique de clé qui autorise des autorisations, sauf si vous utilisez des conditions pour limiter la stratégie de clé. Un astérisque indique chaque identité associée à chaque Compte AWS autorisation d'utilisation de la clé KMS, sauf si une autre déclaration de politique le nie explicitement. Les

utilisateurs des autres utilisateurs Comptes AWS peuvent utiliser votre clé KMS chaque fois qu'ils disposent des autorisations correspondantes sur leur propre compte.

Vous devez également décider quelles autorisations vous souhaitez accorder au compte externe. Par exemple, vous pouvez accorder aux utilisateurs l'autorisation de déchiffrer, mais pas de chiffrer, ou l'autorisation d'afficher la clé KMS sans l'utiliser. Pour obtenir la liste des autorisations sur les clés KMS, veuillez consulter AWS KMS autorisations.

Définition de la politique des clés lors de la création d'une clé KMS

Lorsque vous utilisez l'CreateKeyopération pour créer une clé KMS, vous pouvez utiliser son `Policy` paramètre pour spécifier une politique de clé qui autorise un compte externe, ou des utilisateurs et des rôles externes, à utiliser la clé KMS.

Lorsque vous créez une clé KMS dans le AWS Management Console, vous créez également sa politique de clé. Lorsque vous sélectionnez des identités dans les sections Key Administrators (Administrateurs de clé) et Key Users (Utilisateurs de clé), AWS KMS ajoute des instructions de politique pour ces identités à la politique de clé de la clé KMS. La section Key Users (Utilisateurs de clé) vous permet également d'ajouter des comptes externes en tant qu'utilisateurs de clé.

Lorsque vous entrez l'ID de compte d'un compte externe, AWS KMS deux déclarations sont ajoutées à la politique clé. Cette action affecte uniquement la politique de clé. Les utilisateurs et les rôles du compte externe ne peuvent pas utiliser la clé KMS tant que vous n'avez pas attaché de politiques IAM pour leur accorder tout ou une partie de ces autorisations.

La première instruction de politique de clé accorde au compte externe l'autorisation d'utiliser la clé KMS dans les opérations de chiffrement. La deuxième déclaration de politique clé permet au compte externe de créer, de consulter et de révoquer des autorisations sur la clé KMS, mais uniquement lorsque la demande provient d'un AWS service intégré à AWS KMS. Ces autorisations permettent aux autres AWS services qui cryptent les données utilisateur d'utiliser la clé KMS. Ces autorisations sont conçues pour les clés KMS qui chiffrent les données utilisateur dans les services AWS

Étape 2 : ajouter des politiques IAM dans le compte externe

La politique de clé du compte propriétaire de la clé KMS définit la plage valide pour les autorisations. Cependant, les utilisateurs et les rôles du compte externe ne peuvent pas utiliser la clé KMS tant que

vous n'avez pas attaché des politiques IAM qui délèguent ces autorisations ou utilisé des octrois pour gérer l'accès à la clé KMS. Les politiques IAM sont définies dans le compte externe.

Si la politique de clé accorde l'autorisation au compte externe, vous pouvez attacher des politiques IAM à n'importe quel utilisateur ou rôle du compte. Toutefois, si la politique de clé accorde l'autorisation à des utilisateurs ou des rôles spécifiés, la politique IAM peut uniquement accorder ces autorisations à tous les utilisateurs et rôles spécifiés ou à un sous-ensemble. Si une politique IAM accorde l'accès à la clé KMS à d'autres utilisateurs ou rôles externes, cela n'a aucun effet.

La politique de clé limite également les actions dans la politique IAM. La politique IAM peut déléguer tout ou une partie des actions spécifiées dans la politique de clé. Si la politique IAM répertorie les actions qui ne sont pas spécifiées dans la politique de clé, ces autorisations ne sont pas effectives.

L'exemple de politique IAM suivant autorise le principal à utiliser la clé KMS dans le compte `111122223333` pour les opérations de chiffrement. Pour accorder cette autorisation aux utilisateurs et rôles du compte `444455556666`, attachez la politique aux utilisateurs ou rôles du compte `444455556666`.

> **Note**
>
> L'exemple suivant illustre une politique IAM qui permet de partager une clé KMS avec un autre compte. Remplacez les valeurs `Sid`, `Resource` et `Action` de l'exemple par des valeurs valides pour l'utilisation prévue de votre clé KMS.

```
{
    "Sid": "AllowUseOfKeyInAccount111122223333",
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt",
      "kms:ReEncrypt*",
      "kms:GenerateDataKey*",
      "kms:DescribeKey"
    ],
    "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

Notez les informations suivantes sur cette politique :

- Contrairement aux politiques de clé, les instructions de politique IAM ne contiennent pas l'élément `Principal`. Dans les politiques IAM, le principal est l'identité à laquelle la politique est attachée.
- L'élément `Resource` de la politique IAM identifie la clé KMS que le principal peut utiliser. Pour spécifier une clé KMS, ajoutez son ARN de clé à l'élément `Resource`.
- Vous pouvez spécifier plusieurs clés KMS dans l'élément `Resource`. Si vous ne spécifiez pas de clés KMS particulières dans l'élément `Resource`, vous pouvez accorder par inadvertance l'accès à plus de clés KMS que prévu.
- Pour autoriser l'utilisateur externe à utiliser la clé KMS avec des services AWS qui s'intègrent à AWS KMS, vous pouvez avoir besoin d'ajouter des autorisations à la politique de clé ou à la politique IAM. Pour plus de détails, veuillez consulter Autoriser l'utilisation de clés KMS externes avec Services AWS.

Pour plus d'informations sur l'utilisation des politiques IAM, veuillez consulter Politiques IAM.

Autoriser l'utilisation de clés KMS externes avec Services AWS

Vous pouvez autoriser un utilisateur d'un autre compte à utiliser votre clé KMS avec un service intégré à AWS KMS. Par exemple, un utilisateur d'un compte externe peut utiliser votre clé KMS pour chiffrer les objets d'un compartiment Amazon S3 ou pour chiffrer les secrets dans lesquels ils sont stockés. AWS Secrets Manager

La politique de clé doit accorder à l'utilisateur externe ou au compte de l'utilisateur externe l'autorisation d'utiliser la clé KMS. De plus, vous devez attribuer des politiques IAM à l'identité qui autorise l'utilisateur à utiliser Service AWS. Le service peut également exiger que les utilisateurs disposent d'autorisations supplémentaires dans la politique de clé ou la politique IAM. Pour obtenir la liste des autorisations Service AWS requises sur une clé gérée par le client, consultez la rubrique Protection des données dans le chapitre Sécurité du guide de l'utilisateur ou du guide du développeur du service.

Utilisation de clés KMS dans d'autres comptes

Si vous êtes autorisé à utiliser une clé KMS dans un autre Compte AWS, vous pouvez utiliser la clé KMS dans le AWS Management Console AWS SDKs, AWS CLI, et Outils AWS pour PowerShell.

Pour identifier une clé KMS dans un compte différent dans une commande shell ou une demande d'API, utilisez les identificateurs de clé suivants.

- Pour les opérations cryptographiques, et DescribeKeyGetPublicKey, utilisez l'ARN de la clé ou l'alias ARN de la clé KMS.
- Pour CreateGrant, GetKeyRotationStatusListGrants, et RevokeGrant, utilisez l'ARN de la clé KMS.

Si vous entrez uniquement un identifiant de clé ou un nom d'alias, cela AWS suppose que la clé KMS se trouve dans votre compte.

La AWS KMS console n'affiche pas les clés KMS dans les autres comptes, même si vous êtes autorisé à les utiliser. En outre, les listes de clés KMS affichées dans les consoles d'autres services AWS n'incluent pas de clés KMS dans d'autres comptes.

Pour spécifier une clé KMS dans un autre compte de la console d'un AWS service, vous devez saisir l'ARN de la clé ou l'alias ARN de la clé KMS. L'identificateur de clé requis varie en fonction du service et peut différer entre la console de service et ses opérations d'API. Pour plus de détails, consultez la documentation du service.

Contrôler l'accès aux clés multirégionales

Vous pouvez utiliser des clés multi-région dans des scénarios de conformité, de reprise après sinistre et de sauvegarde qui seraient plus complexes avec les clés à région unique. Toutefois, étant donné que les propriétés de sécurité des clés multi-région sont significativement différentes de celles des clés à région unique, nous vous recommandons de faire preuve de prudence lorsque vous autorisez la création, la gestion et l'utilisation de clés multi-région.

> **Note**
>
> Les instructions de politique IAM existantes avec des caractères génériques dans le champ `Resource` s'appliquent désormais à la fois aux clés à région unique et multi-région. Pour les limiter aux clés KMS à région unique ou aux clés multirégionales, utilisez la clé de MultiRegion condition kms :.

Utilisez vos outils d'autorisation pour empêcher la création et l'utilisation de clés multi-région dans tous les scénarios où une clé à région unique suffira. Autorisez les principaux à répliquer une clé multirégionale uniquement dans les régions Régions AWS qui en ont besoin. Donnez l'autorisation pour les clés multi-région uniquement aux principaux qui en ont besoin et uniquement pour les tâches qui en ont besoin.

Vous pouvez utiliser des politiques clés, des politiques IAM et des subventions pour permettre aux principaux IAM de gérer et d'utiliser des clés multirégionales dans votre. Compte AWS Chaque clé multi-région est une ressource indépendante dotée d'un ARN de clé unique et d'une politique de clé. Vous devez établir et maintenir une stratégie de clé pour chaque clé et vous assurer que les stratégies IAM nouvelles et existantes mettent en œuvre votre stratégie d'autorisation.

Pour prendre en charge les clés multirégionales, AWS KMS utilise un rôle lié au service IAM. Ce rôle donne à AWS KMS les autorisations dont il a besoin pour synchroniser les propriétés partagées. Pour de plus amples informations, veuillez consulter Autorisation de synchronisation AWS KMS des clés multirégionales.

Rubriques

- Notions de base sur les autorisations pour les clés multi-région
- Autorisation des administrateurs et des utilisateurs de clés multi-région

Notions de base sur les autorisations pour les clés multi-région

Lors de la conception de politiques de clé et de politiques IAM pour les clés multi-région, tenez compte des principes suivants.

- Politique de clé — Chaque clé multi-région est une ressource de clé KMS indépendante avec sa propre politique de clé. Vous pouvez appliquer la même politique de clé ou une politique de clé différente à chaque clé de l'ensemble des clés multi-région associées. Les politiques clés ne sont pas des propriétés partagées des clés multirégionales. AWS KMS ne copie ni ne synchronise les politiques clés entre les clés multirégionales associées.

  Lorsque vous créez une réplique de clé dans la AWS KMS console, celle-ci affiche la politique de clé actuelle de la clé primaire pour plus de commodité. Vous pouvez utiliser cette politique de clé, la modifier ou la supprimer et la remplacer. Mais même si vous acceptez la politique de clé primaire telle quelle, AWS KMS cela ne synchronise pas les politiques. Par exemple, si vous modifiez la politique de clé de la clé principale, la politique de clé de la clé de réplica reste la même.
- Politique clé par défaut — Lorsque vous créez des clés multirégionales à l'aide `ReplicateKey` des opérations CreateKeyet, la politique clé par défaut est appliquée sauf si vous spécifiez une stratégie clé dans la demande. Il s'agit de la même politique de clé par défaut qui est appliquée aux clés à région unique.
- Politiques IAM — Comme pour toutes les clés KMS, vous pouvez utiliser des politiques IAM pour contrôler l'accès aux clés multi-région uniquement lorsque la politique de clé le permet. Les

politiques IAM s'appliquent à tous Régions AWS par défaut. Cependant, vous pouvez utiliser des clés de condition, telles que aws : RequestedRegion, pour limiter les autorisations à une région particulière.

Pour créer des clés principales et des clés de réplica, les principaux doivent avoir l'autorisation `kms:CreateKey` dans une politique IAM qui s'applique à la région où la clé est créée.

- Subventions : les AWS KMS subventions sont régionales. Chaque octroi autorise l'ajout d'autorisations sur une clé KMS. Vous pouvez utiliser des octrois pour autoriser des autorisations sur une clé principale ou une clé de réplica multi-région. Mais vous ne pouvez pas utiliser un seul octroi pour autoriser des autorisations sur plusieurs clés KMS, même s'il s'agit de clés multi-région associées.
- ARN de clé — Chaque clé multi-région a un ARN de clé unique. La clé ARNs des clés multirégionales associées possède la même partition, le même compte et le même identifiant de clé, mais des régions différentes.

  Pour appliquer une instruction de politique IAM à une clé multi-région particulière, utilisez son ARN de clé ou un modèle d'ARN de clé qui inclut la région. Pour appliquer une instruction de politique IAM à toutes les clés multi-région associées, utilisez un caractère générique (*) dans l'élément Region de l'ARN, comme illustré dans l'exemple suivant.

  ```
  {
    "Effect": "Allow",
    "Action": [
      "kms:Describe*",
      "kms:List*"
    ],
    "Resource": {
        "arn:aws:kms:*::111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab"
    }
  }
  ```

  Pour appliquer une déclaration de politique à toutes les clés multirégionales de votre clé Compte AWS, vous pouvez utiliser la condition de MultiRegion politique kms : ou un modèle d'identification de clé incluant le `mrk-` préfixe distinctif.
- Rôle lié à un service — Les principaux qui créent des clés primaires multirégionales doivent disposer de l'autorisation iam :. CreateServiceLinkedRole

Pour synchroniser les propriétés partagées des clés multirégionales associées, AWS KMS assume un rôle lié à un service IAM. AWS KMS crée le rôle lié au service Compte AWS chaque fois que vous créez une clé primaire multirégionale. (Si le rôle existe, AWS KMS le recrée, ce qui n'a aucun effet nocif.) Le rôle est valable dans toutes les régions. Pour permettre AWS KMS de créer (ou de recréer) le rôle lié au service, les principaux qui créent des clés primaires multirégionales doivent disposer de l'autorisation iam :. CreateServiceLinkedRole

Autorisation des administrateurs et des utilisateurs de clés multi-région

Les principaux qui créent et gèrent des clés multi-région ont besoin des autorisations suivantes dans les régions principale et de réplica :

- `kms:CreateKey`
- `kms:ReplicateKey`
- `kms:UpdatePrimaryRegion`
- `iam:CreateServiceLinkedRole`

Création d'une clé principale

Pour créer une clé primaire multirégionale, le principal a besoin des CreateServiceLinkedRole autorisations kms : CreateKey et iam : dans le cadre d'une politique IAM effective dans la région de la clé primaire. Les principaux qui disposent de ces autorisations peuvent créer des clés à région unique et multi-région à moins que vous ne restreigniez leurs autorisations.

L'`iam:CreateServiceLinkedRole`autorisation permet AWS KMS de créer le AWSServiceRoleForKeyManagementServiceMultiRegionKeysrôle pour synchroniser les propriétés partagées des clés multirégionales associées.

Par exemple, cette politique IAM permet à un principal de créer n'importe quel type de clé KMS.

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action": [
        "kms:CreateKey",
        "iam:CreateServiceLinkedRole"
      ],
```

```
      "Effect":"Allow",
      "Resource":"*"
  }
}
```

Pour autoriser ou refuser l'autorisation de créer des clés primaires multirégionales, utilisez la clé de MultiRegion condition kms :. Les valeurs valides sont `true` (clé multi-région) ou `false` (clé à région unique). Par exemple, l'instruction de politique IAM utilise une action `Deny` avec la clé de condition `kms:MultiRegion` pour empêcher les principaux de créer des clés multi-région.

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action":"kms:CreateKey",
      "Effect":"Deny",
      "Resource":"*",
      "Condition": {
          "Bool": "kms:MultiRegion": true
      }
  }
}
```

Réplication de clés

Pour créer une clé de réplica multi-région, le principal a besoin des autorisations suivantes :

- kms : ReplicateKey autorisation dans la politique de clé de la clé primaire.
- kms : CreateKey autorisation dans une politique IAM en vigueur dans la région de la clé de réplique.

Soyez prudent lorsque vous autorisez ces autorisations. Elles permettent aux principaux de créer des clés KMS et les politiques de clé qui autorisent leur utilisation. L'autorisation `kms:ReplicateKey` permet également le transfert d'éléments de clé au-delà des limites de la région dans AWS KMS.

Pour limiter les limites Régions AWS dans lesquelles une clé multirégionale peut être répliquée, utilisez la clé de ReplicaRegion condition kms :. Elle ne limite que l'autorisation `kms:ReplicateKey`. Sinon, elle n'a aucun effet. Par exemple, la politique de clé suivante autorise le principal à répliquer cette clé principale, mais uniquement dans les régions spécifiées.

```
{
```

```
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Administrator"
  },
  "Action": "kms:ReplicateKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ReplicaRegion": [
         "us-east-1",
         "eu-west-3",
         "ap-southeast-2"
      ]
    }
  }
}
```

Mise à jour de la région principale

Les principaux autorisés peuvent transformer une clé de réplica en clé principale, ce qui transforme l'ancienne clé principale en un réplica. Cette action s'appelle la mise à jour de la région principale. Pour mettre à jour la région principale, le principal a besoin d'une UpdatePrimaryRegion autorisation de km : dans les deux régions. Vous pouvez fournir ces autorisations dans une politique de clé ou une politique IAM.

- `kms:UpdatePrimaryRegion` sur la clé principale. Cette autorisation doit être effective dans la région de clé principale.
- `kms:UpdatePrimaryRegion` sur la clé de réplica. Cette autorisation doit être effective dans la région de clé de réplica.

Par exemple, la politique de clé suivante donne aux utilisateurs qui peuvent endosser le rôle Administrateur l'autorisation de mettre à jour la région principale de la clé KMS. Cette clé KMS peut être la clé principale ou une clé de réplica dans cette opération.

```
{
  "Effect": "Allow",
  "Resource": "*",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Administrator"
  },
```

```
  "Action": "kms:UpdatePrimaryRegion"
}
```

Pour restreindre la Régions AWS capacité d'héberger une clé primaire, utilisez la clé de PrimaryRegion condition kms :. Par exemple, la déclaration de politique IAM suivante permet aux principaux de mettre à jour la région principale des clés multirégionales dans le Compte AWS, mais uniquement lorsque la nouvelle région principale est l'une des régions spécifiées.

```
{
  "Effect": "Allow",
  "Action": "kms:UpdatePrimaryRegion",
  "Resource": {
      "arn:aws:kms:*:111122223333:key/*"
  },
  "Condition": {
    "StringEquals": {
      "kms:PrimaryRegion": [
         "us-west-2",
         "sa-east-1",
         "ap-southeast-1"
      ]
    }
  }
}
```

Utilisation et gestion des clés multi-région

Par défaut, les principaux qui ont l'autorisation d'utiliser et de gérer les clés KMS dans un Compte AWS et une région ont également l'autorisation d'utiliser et de gérer des clés multi-région. Cependant, vous pouvez utiliser la clé de MultiRegion condition kms : pour autoriser uniquement les clés à région unique ou uniquement les clés multirégionales. Vous pouvez également utiliser la clé de MultiRegionKeyType condition kms : pour autoriser uniquement les clés primaires multirégionales ou uniquement les clés de réplique. Les deux clés de condition contrôlent l'accès à l'CreateKeyopération et à toute opération utilisant une clé KMS existante, telle que Encrypt ou EnableKey.

L'exemple suivant d'instruction de politique IAM utilise la clé de condition `kms:MultiRegion` pour empêcher les principaux d'utiliser ou de gérer une clé multi-région.

```
{
  "Effect": "Deny",
```

```
  "Action": "kms:*",
  "Resource": "*",
  "Condition": {
    "Bool": "kms:MultiRegion": true
  }
}
```

Cet exemple d'instruction de politique IAM utilise la condition `kms:MultiRegionKeyType` pour permettre aux principaux de planifier et d'annuler la suppression de clé, mais uniquement sur les clés de réplica multi-région.

```
{
  "Effect": "Allow",
  "Action": [
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": {
      "arn:aws:kms:us-west-2:111122223333:key/*"
  },
  "Condition": {
    "StringEquals": "kms:MultiRegionKeyType": "REPLICA"
  }
}
```

Déterminer l'accès à AWS KMS keys

Pour déterminer l'étendue complète de qui ou de quoi a actuellement accès à une AWS KMS key, vous devez examiner la politique clé de la clé KMS, toutes les autorisations qui s'appliquent à la clé KMS et potentiellement toutes les politiques AWS Identity and Access Management (IAM). Vous pouvez le faire pour déterminer la portée de l'utilisation potentielle d'une clé KMS ou pour mieux répondre aux exigences d'audit ou de conformité. Les rubriques suivantes peuvent vous aider à générer une liste complète des principaux AWS (identités) qui ont actuellement accès à une clé KMS.

Rubriques

- Examen de la politique de clé
- Examen des politiques IAM
- Examen des octrois

Examen de la politique de clé

Les politiques de clé constituent le principal moyen de contrôler l'accès aux clés KMS. Chaque clé KMS a exactement une politique de clé.

Lorsqu'une politique de clé inclut la politique de clé par défaut, elle permet aux administrateurs IAM du compte d'utiliser des politiques IAM pour contrôler l'accès à la clé KMS. En outre, si la politique de clé donne à un autre Compte AWS l'autorisation d'utiliser la clé KMS, les administrateurs IAM du compte externe peuvent utiliser des politiques IAM pour déléguer ces autorisations. Pour déterminer la liste complète des principaux qui peuvent accéder à la clé KMS, examinez les politiques IAM.

Pour consulter la politique clé d'une clé gérée par le AWS KMS client ou Clé gérée par AWSde votre compte, utilisez l'GetKeyPolicyopération AWS Management Console ou dans l' AWS KMS API. Pour afficher la politique de clé, vous devez disposer des autorisations `kms:GetKeyPolicy` pour la clé KMS. Pour obtenir des instructions sur l'affichage de la politique de clé d'une clé KMS, veuillez consulter the section called “Afficher une politique clé”.

Examinez le document de politique de clé et notez tous les principaux spécifiés dans l'élément `Principal` de chaque instruction de politique. Dans une déclaration de politique ayant un `Allow` effet, les utilisateurs IAM, les rôles IAM et Comptes AWS l'`Principal`élément ont accès à cette clé KMS.

> **Note**
>
> Ne définissez pas le principal sur un astérisque (*) dans une instruction de politique de clé qui autorise des autorisations, sauf si vous utilisez des conditions pour limiter la politique de clé. Un astérisque indique chaque identité associée à chaque Compte AWS autorisation d'utilisation de la clé KMS, sauf si une autre déclaration de politique le nie explicitement. Les utilisateurs des autres utilisateurs Comptes AWS peuvent utiliser votre clé KMS chaque fois qu'ils disposent des autorisations correspondantes sur leur propre compte.

Les exemples suivants utilisent les instructions de politique trouvées dans la politique de clé par défaut pour montrer comment procéder.

Example Instruction de politique 1

```
{
  "Sid": "Enable IAM User Permissions",
```

```
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
  "Action": "kms:*",
  "Resource": "*"
}
```

Dans l'énoncé de politique 1, `arn:aws:iam::111122223333:root` un principal de AWS compte fait référence au Compte AWS 111122223333. (Il ne s'agit pas de l'utilisateur root du compte). Par défaut, une déclaration de politique comme celle-ci est incluse dans le document de politique clé lorsque vous créez une nouvelle clé KMS avec le AWS Management Console, ou lorsque vous créez une nouvelle clé KMS par programmation sans fournir de politique clé.

Document de politique clé contenant une déclaration autorisant l'accès aux politiques d' Compte AWS activation IAM du compte afin d'autoriser l'accès à la clé KMS. Cela signifie que les utilisateurs et rôles figurant dans le compte peuvent avoir accès à la clé KMS, même s'ils ne sont pas répertoriés explicitement en tant que principaux dans le document de politique de clé. Prenez soin d'examiner toutes les politiques IAM Comptes AWS répertoriées comme principales afin de déterminer si elles autorisent l'accès à cette clé KMS.

Example Instruction de politique 2

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/KMSKeyAdmins"},
  "Action": [
    "kms:Describe*",
    "kms:Put*",
    "kms:Create*",
    "kms:Update*",
    "kms:Enable*",
    "kms:Revoke*",
    "kms:List*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

Dans la déclaration de politique 2, `arn:aws:iam::111122223333:role/KMSKeyAdmins` fait référence au rôle IAM nommé KMSKey Admins dans Compte AWS le document 111122223333. Les utilisateurs autorisés à assumer ce rôle sont habilités à effectuer les opérations répertoriées dans l'instruction de politique, à savoir les opérations administratives permettant de gérer une clé KMS.

Example Instruction de politique 3

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},
  "Action": [
    "kms:DescribeKey",
    "kms:GenerateDataKey*",
    "kms:Encrypt",
    "kms:ReEncrypt*",
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

Dans la déclaration de politique 3, `arn:aws:iam::111122223333:role/EncryptionApp` fait référence au rôle IAM nommé EncryptionApp dans Compte AWS 111122223333. Les principaux autorisés à assumer ce rôle sont habilités à effectuer les opérations répertoriées dans l'instruction de politique, y compris les opérations de chiffrement relatives à une clé KMS de chiffrement symétrique.

Example Instruction de politique 4

```
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},
  "Action": [
    "kms:ListGrants",
    "kms:CreateGrant",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

Dans la déclaration de politique 4, `arn:aws:iam::111122223333:role/EncryptionApp` fait référence au rôle IAM nommé EncryptionApp dans Compte AWS 111122223333. Les principaux autorisés à assumer ce rôle sont habilités à effectuer les opérations répertoriées dans l'instruction de politique. Ces actions, lorsqu'elles sont combinées aux actions autorisées dans l'exemple d'instruction de politique 3, sont celles requises pour déléguer l'utilisation de la clé KMS à la plupart des services AWS qui s'intègrent à AWS KMS, notamment aux services qui utilisent des octrois. La GrantIsFor AWSResource valeur kms : dans l'`Condition`élément garantit que la délégation n'est autorisée que lorsque le délégué est un AWS service qui intègre AWS KMS et utilise des autorisations d'autorisation.

Pour découvrir toutes les différentes façons de spécifier un principal dans un document de politique de clé, veuillez consulter la page Spécification d'un principal dans le Guide de l'utilisateur IAM.

Pour en savoir plus sur les AWS KMS principales politiques, voirPolitiques clés en AWS KMS.

Examen des politiques IAM

Outre la politique de clé et les octrois, vous pouvez utiliser des politiques IAM pour autoriser l'accès à une clé KMS. Pour plus d'informations sur la manière dont les politiques de clé et les politiques IAM fonctionnent ensemble, veuillez consulter AWS KMS Permissions de résolution des.

Pour déterminer quels principaux ont actuellement accès à une clé KMS via les politiques IAM, vous pouvez utiliser l'outil simulateur de politiques IAM basé sur le navigateur ou vous pouvez adresser des demandes à l'API IAM.

Manières d'examiner les politiques IAM

- Examen des politiques IAM avec le simulateur de politiques IAM
- Examen des politiques IAM avec l'API IAM

Examen des politiques IAM avec le simulateur de politiques IAM

Le simulateur de politiques IAM peut vous aider à découvrir quels principaux ont accès à une clé KMS via une politique IAM.

Pour utiliser le simulateur de politiques IAM pour déterminer l'accès à une clé KMS

1. Connectez-vous au simulateur de politique IAM, AWS Management Console puis ouvrez-le à https://policysim.aws.amazon.com/ l'adresse.

2. Dans le volet Users, Groups, and Roles, choisissez l'utilisateur, le groupe ou le rôle dont vous souhaitez simuler les politiques.
3. (Facultatif) Décochez les cases en regard de toutes les politiques que vous souhaitez ignorer pour la simulation. Pour simuler toutes les politiques, laissez toutes les politiques sélectionnées.
4. Dans le volet Policy Simulator, procédez comme suit :
   a. Pour Select service, choisissez Key Management Service.
   b. Pour simuler des AWS KMS actions spécifiques, pour Sélectionner des actions, choisissez les actions à simuler. Pour simuler toutes les AWS KMS actions, choisissez Tout sélectionner.
5. (Facultatif) Le simulateur de politiques simule l'accès à toutes les clés KMS par défaut. Pour simuler l'accès à une clé KMS spécifique, sélectionnez Simulation Settings (Paramètres de simulation), puis saisissez l'Amazon Resource Name (ARN) de la clé KMS à simuler.
6. Choisissez Exécuter la simulation.

Vous pouvez afficher les résultats de la simulation dans la section Résultats. Répétez les étapes 2 à 6 pour chaque utilisateur, groupe et rôle figurant dans le Compte AWS.

Examen des politiques IAM avec l'API IAM

Vous pouvez utiliser l'API IAM pour examiner par programmation les politiques IAM. Les étapes suivantes offrent une présentation générale de la façon de procéder :

1. Pour chaque utilisateur Compte AWS répertorié en tant que principal dans la politique clé (c'est-à-dire chaque principal de AWS compte spécifié dans ce format :`"Principal": {"AWS": "arn:aws:iam::111122223333:root"}`), utilisez les ListRolesopérations ListUserset de l'API IAM pour obtenir tous les utilisateurs et rôles du compte.
2. Pour chaque utilisateur et chaque rôle de la liste, utilisez l'SimulatePrincipalPolicyopération de l'API IAM en transmettant les paramètres suivants :
   - Pour `PolicySourceArn`, spécifiez le nom ARN (Amazon Resource Name) d'un utilisateur ou d'un rôle figurant dans votre liste. Vous ne pouvez spécifier qu'un seul nom `PolicySourceArn` pour chaque demande `SimulatePrincipalPolicy`. Vous devez donc appeler cette opération plusieurs fois, une fois pour chaque utilisateur et rôle de votre liste.
   - Pour la `ActionNames` liste, spécifiez chaque action d' AWS KMS API à simuler. Pour simuler toutes les actions de AWS KMS l'API, utilisez`kms:*`. Pour tester des actions AWS KMS d'API individuelles, faites précéder chaque action d'API de `kms:` « », par exemple

« `kms:ListKeys` ». Pour obtenir la liste complète des actions de l'API AWS KMS , consultez Actions dans la référence d'API AWS Key Management Service .

- (Facultatif) Pour déterminer si les utilisateurs ou les rôles ont accès à des clés KMS spécifiques, utilisez le `ResourceArns` paramètre pour spécifier une liste des Amazon Resource Names (ARNs) des clés KMS. Pour déterminer si les utilisateurs ou les rôles ont accès à une clé KMS quelconque, omettez le paramètre `ResourceArns`.

IAM répond à chaque demande `SimulatePrincipalPolicy` avec une décision d'évaluation : `allowed`, `explicitDeny` ou `implicitDeny`. Pour chaque réponse contenant une décision d'évaluation de`allowed`, la réponse inclut le nom de l'opération d' AWS KMS API spécifique autorisée. Elle inclut également l'ARN de la clé KMS qui a été utilisée dans l'évaluation, le cas échéant.

Examen des octrois

Les subventions sont des mécanismes avancés permettant de spécifier les autorisations que vous ou un AWS service intégré AWS KMS pouvez utiliser pour spécifier comment et quand une clé KMS peut être utilisée. Les octrois sont attachés à une clé KMS et chaque octroi contient le principal qui reçoit l'autorisation d'utiliser la clé KMS et la liste des opérations autorisées. Les octrois représentent une alternative à la politique de clé et sont utiles pour des cas d'utilisation spécifiques. Pour de plus amples informations, veuillez consulter Subventions en AWS KMS.

Pour obtenir une liste des autorisations pour une clé KMS, utilisez l' AWS KMS ListGrantsopération. Vous pouvez examiner les octrois définis pour une clé KMS afin de déterminer qui a actuellement accès à la clé KMS pour l'utiliser via ces octrois. Par exemple, ce qui suit est une représentation JSON d'un octroi qui a été obtenu à partir de la commande list-grants dans l' AWS CLI.


```
{"Grants": [{
  "Operations": ["Decrypt"],
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "Name": "0d8aa621-43ef-4657-b29c-3752c41dc132",
  "RetiringPrincipal": "arn:aws:iam::123456789012:root",
  "GranteePrincipal": "arn:aws:sts::111122223333:assumed-role/aws:ec2-infrastructure/
i-5d476fab",
  "GrantId": "dc716f53c93acacf291b1540de3e5a232b76256c83b2ecb22cdefa26576a2d3e",
  "IssuingAccount": "arn:aws:iam::111122223333:root",
  "CreationDate": 1.444151834E9,
  "Constraints": {"EncryptionContextSubset": {"aws:ebs:id": "vol-5cccfb4e"}}
```

```
}]}
```

Pour identifier qui a accès à la clé KMS pour l'utiliser, recherchez l'élément `"GranteePrincipal"`. Dans l'exemple précédent, le bénéficiaire principal est un utilisateur du rôle supposé associé à l' EC2 instance i-5d476fab. L' EC2infrastructure utilise ce rôle pour attacher le volume EBS chiffré vol-5cccfb4e à l'instance. Dans ce cas, le rôle EC2 d'infrastructure est autorisé à utiliser la clé KMS car vous avez précédemment créé un volume EBS chiffré protégé par cette clé KMS. Vous avez ensuite attaché le volume à une EC2 instance.

Ce qui suit est un autre exemple de représentation JSON d'un octroi qui a été obtenu à partir de la commande list-grants dans l' AWS CLI. Dans l'exemple suivant, le directeur du bénéficiaire est un autre Compte AWS.

```
{"Grants": [{
  "Operations": ["Encrypt"],
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "Name": "",
  "GranteePrincipal": "arn:aws:iam::444455556666:root",
  "GrantId": "f271e8328717f8bde5d03f4981f06a6b3fc18bcae2da12ac38bd9186e7925d11",
  "IssuingAccount": "arn:aws:iam::111122223333:root",
  "CreationDate": 1.444151269E9
}]}
```

Contexte de chiffrement

> **Note**
>
> Vous ne pouvez pas spécifier de contexte de chiffrement dans une opération de chiffrement avec une clé KMS asymétrique ou une clé KMS HMAC. Les algorithmes asymétriques et les algorithmes MAC ne prennent pas en charge un contexte de chiffrement.

Toutes les opérations de AWS KMS chiffrement avec des clés KMS de chiffrement symétrique acceptent un contexte de chiffrement, un ensemble facultatif de paires clé-valeur qui peut contenir des informations contextuelles supplémentaires sur les données. Vous pouvez insérer un contexte de chiffrement dans les `Encrypt` opérations afin AWS KMS d'améliorer l'autorisation et l'auditabilité de vos appels de déchiffrement d' AWS KMS API. AWS KMS utilise le contexte de chiffrement en tant que données authentifiées supplémentaires (AAD) pour prendre en charge le chiffrement

authentifié. Le contexte de chiffrement est lié cryptographiquement au texte chiffré, de sorte que le même contexte de chiffrement est requis pour déchiffrer les données.

Le contexte de chiffrement n'est pas secret ou chiffré. Il apparaît en texte brut dans les journaux AWS CloudTrail pour vous permettre d'identifier et de classer vos opérations de chiffrement. Votre contexte de chiffrement ne doit pas inclure d'informations sensibles. Nous recommandons que votre chiffrement le contexte décrive les données en cours de chiffrement ou de déchiffrement. Par exemple, lorsque vous chiffrez un fichier, vous pouvez utiliser une partie du chemin de fichier comme contexte de chiffrement.

```
"encryptionContext": {
    "department": "10103.0"
}
```

Par exemple, lorsque vous chiffrez des volumes et des instantanés créés avec l'opération Amazon Elastic Block Store (Amazon EBS) CreateSnapshot, Amazon EBS utilise l'ID de volume comme valeur de contexte de chiffrement.

```
"encryptionContext": {
  "aws:ebs:id": "vol-abcde12345abc1234"
}
```

Vous pouvez également utiliser le contexte de chiffrement pour affiner ou limiter l'accès aux AWS KMS keys dans votre compte. Vous pouvez utiliser le contexte de chiffrement en tant que contrainte dans les octrois et en tant que condition dans les instructions de politique.

Pour savoir comment utiliser le contexte de chiffrement pour protéger l'intégrité des données chiffrées, veuillez consulter l'article Comment protéger l'intégrité de vos données chiffrées à l'aide de AWS Key Management Service et EncryptionContext sur le blog sur la AWS sécurité.

Règles liées au contexte de chiffrement

AWS KMS applique les règles suivantes pour les valeurs et les clés de contexte de chiffrement.

- La clé et la valeur d'une paire de contexte de chiffrement doivent être des chaînes littérales simples. Si vous utilisez un type différent, tel qu'un nombre entier ou à virgule flottante, AWS KMS l'interprète comme une chaîne.
- Les clés et les valeurs dans un contexte de chiffrement peuvent inclure des caractères Unicode. Si un contexte de chiffrement inclut des caractères non autorisés dans les politiques

de clé ou les politiques IAM, vous ne pourrez pas spécifier le contexte de chiffrement dans les clés de condition de politique, telles que `kms:EncryptionContext:context-key` et `kms:EncryptionContextKeys`. Pour plus d'informations sur les règles de document de politique de clé, voir Format de politique de clé. Pour plus d'informations sur les règles du document de politique IAM, veuillez consulter Exigences relatives aux noms IAM dans le Guide de l'utilisateur IAM.

Contexte de chiffrement dans les politiques

Le contexte de chiffrement est utilisé principalement pour vérifier l'intégrité et l'authenticité. Mais vous pouvez également utiliser le contexte de chiffrement pour contrôler l'accès au chiffrement symétrique AWS KMS keys dans les politiques IAM et les politiques clés.

Les clés de EncryptionContextKeys condition kms EncryptionContext : et kms : accordent (ou refusent) une autorisation uniquement lorsque la demande inclut des clés de contexte de chiffrement ou des paires clé-valeur particulières.

Par exemple, l'instruction de politique de clé suivante autorise le rôle `RoleForExampleApp` à utiliser la clé KMS dans les opérations `Decrypt`. Elle utilise la clé de condition `kms:EncryptionContext:context-key` pour accorder cette autorisation uniquement lorsque le contexte de chiffrement de la demande inclut une paire de contexte de chiffrement `AppName:ExampleApp`.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:AppName": "ExampleApp"
    }
  }
}
```

Pour plus d'informations sur ces clés de condition de contexte de chiffrement, consultez Clés de condition pour AWS KMS.

Contexte de chiffrement dans des octrois

Lorsque vous créez une subvention, vous pouvez inclure des contraintes de subvention qui établissent les conditions des autorisations d'octroi. AWS KMS prend en charge deux contraintes d'octroi`EncryptionContextSubset`, qui impliquent toutes les deux le contexte de chiffrement dans une demande d'opération de chiffrement. `EncryptionContextEquals` Lorsque vous utilisez ces contraintes d'octroi, les autorisations de l'octroi sont effectives uniquement lorsque le contexte de chiffrement de la demande pour l'opération de chiffrement satisfait aux exigences des contraintes d'octroi.

Par exemple, vous pouvez ajouter une `EncryptionContextEquals` contrainte d'octroi qui autorise l'GenerateDataKeyopération. Avec cette contrainte, l'octroi autorise l'opération uniquement lorsque le contexte de chiffrement de la demande est une correspondance sensible à la casse pour le contexte de chiffrement de la contrainte d'octroi.

```
$ aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:user/exampleUser \
    --retiring-principal arn:aws:iam::111122223333:role/adminRole \
    --operations GenerateDataKey \
    --constraints EncryptionContextEquals={Purpose=Test}
```

Une demande telle que la suivante émanant du principal bénéficiaire satisferait à la contrainte `EncryptionContextEquals`.

```
$ aws kms generate-data-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-spec AES_256 \
    --encryption-context Purpose=Test
```

Pour plus d'informations sur les contraintes d'octroi, veuillez consulter Utilisation des contraintes d'octroi. Pour de plus amples informations sur les octrois, veuillez consulter the section called “Octrois”.

Consignation du contexte de chiffrement

AWS KMS utilise AWS CloudTrail pour consigner le contexte de chiffrement et vous permettre de déterminer les clés KMS et les données qui ont été consultées. L'entrée de journal montre

exactement quelles clés KMS ont été utilisées pour chiffrer ou déchiffrer les données spécifiques référencées par le contexte de chiffrement dans l'entrée de journal.

> **Important**
>
> Etant donné que le contexte de chiffrement est consigné, il ne doit contenir d'informations sensibles.

Stockage du contexte de chiffrement

Pour simplifier l'utilisation de n'importe quel contexte de chiffrement lorsque vous appelez les opérations `Decrypt` ou `ReEncrypt`, vous pouvez stocker le contexte de chiffrement en même temps que les données chiffrées. Nous vous conseillons de stocker juste assez du contexte de chiffrement pour créer aisément le contexte de chiffrement complet, lorsque cela est nécessaire pour le chiffrement ou le déchiffrement.

Par exemple, si le contexte de chiffrement est le chemin d'accès complet à un fichier, stockez uniquement une partie de ce chemin d'accès avec le contenu du fichier chiffré. Ensuite, lorsque vous aurez besoin du contexte de chiffrement complet, reconstruisez-le à partir du fragment stocké. En cas de tentative d'accès non autorisé au fichier, par exemple pour le renommer ou le déplacer, la valeur du contexte de chiffrement change et la requête de déchiffrement échoue.

Test des autorisations

Pour l'utiliser AWS KMS, vous devez disposer d'informations d'identification AWS permettant d'authentifier vos demandes d'API. Les informations d'identification doivent inclure des autorisations pour accéder aux clés KMS et aux alias. Les autorisations sont déterminées par les stratégies de clé, les politiques IAM, les octrois et les contrôles d'accès intercomptes. Outre le contrôle de l'accès aux clés KMS, vous pouvez contrôler l'accès à votre CloudHSM et à vos magasins de clés personnalisés.

Vous pouvez spécifier le paramètre d’API `DryRun` pour vérifier que vous disposez des autorisations nécessaires pour utiliser les clés AWS KMS . Vous pouvez également l'utiliser `DryRun` pour vérifier que les paramètres de demande dans un appel d' AWS KMS API sont correctement spécifiés.

Rubriques

- Quel est le DryRun paramètre ?
- Spécification DryRun à l'aide de l'API

Quel est le DryRun paramètre ?

`DryRun` est un paramètre d'API facultatif que vous spécifiez pour vérifier que les appels d'API AWS KMS aboutiront. Utilisez `DryRun` pour tester votre appel d'API, avant de passer réellement l'appel à AWS KMS. Vous pouvez modifier les valeurs suivantes :

- Que vous disposez des autorisations nécessaires pour utiliser les clés AWS KMS .
- Que vous avez correctement spécifié les paramètres lors de l'appel.

AWS KMS prend en charge l'utilisation du `DryRun` paramètre dans certaines actions d'API :

- CreateGrant
- Decrypt (Déchiffrer)
- DeriveSharedSecret
- Encrypt (Chiffrer)
- GenerateDataKey
- GenerateDataKeyPair
- GenerateDataKeyPairWithoutPlaintext
- GenerateDataKeyWithoutPlaintext
- GenerateMac
- ReEncrypt
- RetireGrant
- RevokeGrant
- Sign (Signer)
- Verify (Vérifier)
- VerifyMac

L'utilisation du paramètre `DryRun` entraînera des frais et sera facturée comme une demande d'API standard. Pour plus d'informations sur la AWS KMS tarification, consultez la section AWS Key Management Service Tarification.

Toutes les demandes d'API utilisant le paramètre `DryRun` s'appliquent au quota de demandes de l'API et peuvent entraîner une exception de limitation si vous dépassez un quota de demandes d'API.

Par exemple, le fait d'appeler Decrypt avec `DryRun` ou sans `DryRun` compte pour le même quota d'opérations cryptographiques. Pour en savoir plus, veuillez consulter Limitation des demandes AWS KMS.

Chaque appel à une opération d' AWS KMS API est capturé en tant qu'événement et enregistré dans un AWS CloudTrail journal. Le résultat de toutes les opérations qui spécifient le `DryRun` paramètre apparaît dans votre CloudTrail journal. Pour de plus amples informations, veuillez consulter Journalisation des appels d' AWS KMS API avec AWS CloudTrail.

Spécification DryRun à l'aide de l'API

Pour l'utiliser`DryRun`, spécifiez le `–dry-run` paramètre dans AWS CLI les commandes et les appels d' AWS KMS API qui le prennent en charge. Lorsque vous le ferez, AWS KMS nous vérifierons si votre appel aboutira. AWS KMS les appels utilisés `DryRun` échoueront toujours et renverront un message contenant des informations sur la raison de l'échec de l'appel. Le message peut inclure les exceptions suivantes :

- `DryRunOperationException` - La demande aboutirait si `DryRun` n'était pas spécifié.
- `ValidationException` - La demande n'a pas réussi à spécifier un paramètre d'API incorrect.
- `AccessDeniedException` - Vous ne disposez pas des autorisations pour exécuter l'action d'API spécifiée sur la ressource KMS.

Par exemple, la commande suivante utilise l'CreateGrantopération et crée une autorisation qui permet aux utilisateurs autorisés à assumer le `keyUserRole` rôle d'appeler l'opération de déchiffrement sur une clé KMS symétrique spécifiée. Le paramètre `DryRun` est spécifié.

```
$  aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \
    --operations Decrypt \
    --dry-run
```

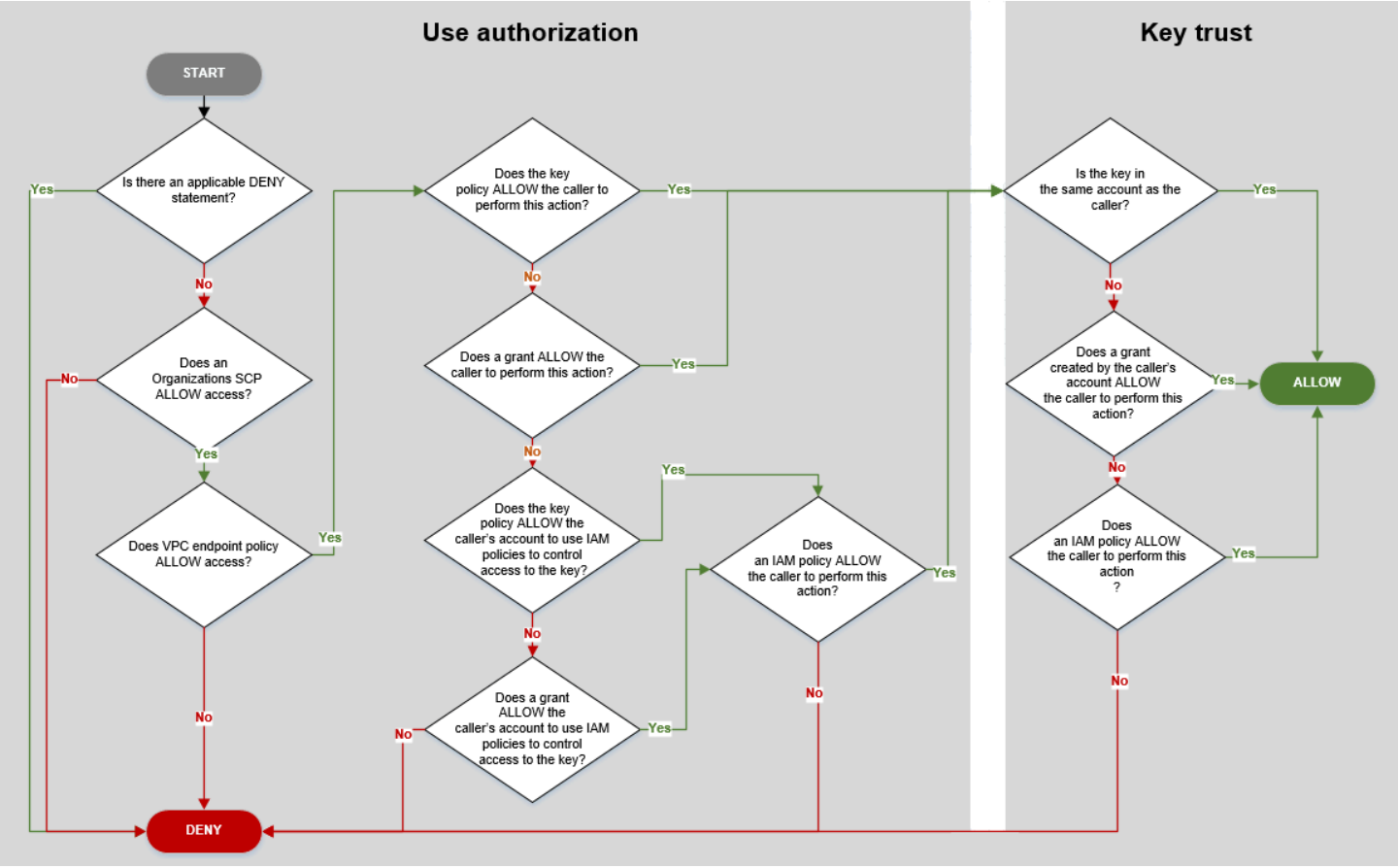
AWS KMS Permissions de résolution des

Lorsque vous autorisez l'accès à une clé KMS, AWS KMS évalue les points suivants :

- La politique de clé attachée à la clé KMS. La politique clé est toujours définie dans la région Compte AWS et qui possède la clé KMS.

- Toutes les politiques IAM attribuées à l'utilisateur ou au rôle à l'origine de la demande. Les politiques IAM qui régissent l'utilisation d'une clé KMS par un principal sont toujours définies dans le Compte AWS du principal.
- Tous les octrois qui s'appliquent à la clé KMS.
- D'autres types de politiques qui peuvent s'appliquer à la demande d'utilisation de la clé KMS, tels que les politiques de contrôle des services AWS Organizations et les politiques de point de terminaison d'un VPC. Ces politiques sont facultatives et autorisent toutes les actions par défaut, mais vous pouvez les utiliser pour restreindre les autorisations accordées par ailleurs aux principaux.

AWS KMS évalue ensemble ces mécanismes de politique afin de déterminer si l'accès à la clé KMS est autorisé ou refusé. Pour ce faire, AWS KMS utilise un processus similaire à celui décrit dans l'organigramme suivant. Le diagramme suivant fournit une représentation visuelle du processus d'évaluation de la politique.



Ce diagramme est divisé en deux parties. Les parties semblent être séquentielles, mais elles sont généralement évaluées en même temps.

- Use authorization détermine si vous êtes autorisé à utiliser une clé KMS en fonction de sa politique de clé, des politiques IAM, des octrois et des autres politiques applicables.
- Key trust détermine si vous devez approuver une clé KMS que vous êtes autorisé à utiliser. En général, vous faites confiance aux ressources de votre Compte AWS. Mais vous pouvez également être sûr d'utiliser les clés KMS d'une autre manière Compte AWS si une licence ou une politique IAM de votre compte vous autorise à utiliser la clé KMS.

Vous pouvez utiliser ce diagramme de flux pour découvrir pourquoi un appelant est autorisé ou non à utiliser une clé KMS. Vous pouvez également l'utiliser pour évaluer vos politiques et octrois. Par exemple, le diagramme montre qu'un appelant peut se voir refuser l'accès par une instruction `DENY` explicite, ou en l'absence d'une instruction `ALLOW` explicite, dans la politique de clé, la politique IAM, ou l'octroi.

Le diagramme peut expliquer certains scénarios d'autorisation courants.

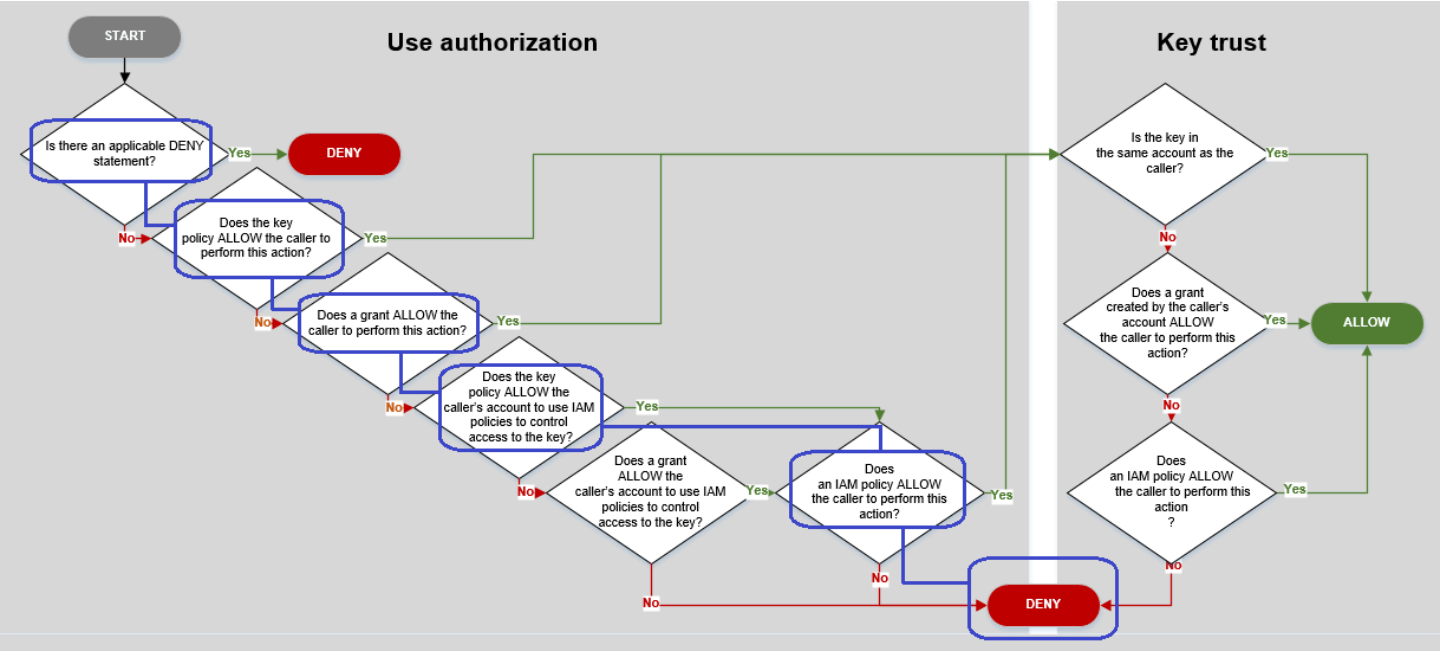
Exemples d'autorisation

- Exemple 1 : L'utilisateur se voit refuser l'accès à une clé KMS dans son Compte AWS
- Exemple 2 : L'utilisateur assume un rôle autorisé à utiliser une clé KMS dans un autre Compte AWS

Exemple 1 : L'utilisateur se voit refuser l'accès à une clé KMS dans son Compte AWS

Alice est une utilisatrice IAM du Compte AWS 111122223333. L'accès à une clé KMS lui a été refusé sur ce même Compte AWS. Pourquoi Alice ne peut-elle pas utiliser la clé KMS ?

Dans ce cas, Alice se voit refuser l'accès à la clé KMS, car aucune politique de clé, aucune politique IAM ou aucun octroi ne lui accorde les autorisations requises. La politique clé de la clé KMS permet d' Compte AWS utiliser des politiques IAM pour contrôler l'accès à la clé KMS, mais aucune politique IAM n'autorise Alice à utiliser la clé KMS.



Considérez les politiques appropriées dans cet exemple.

- La clé KMS qu'Alice souhaite utiliser dispose de la politique de clé par défaut. Cette politique autorise le Compte AWS qui possède la clé KMS à utiliser des politiques IAM pour contrôler l'accès à la clé KMS. Cette politique de clé remplit la condition La politique de clé AUTORISE-t-elle le compte du principal à utiliser les politiques IAM pour contrôler l'accès à la clé ? du diagramme de flux.

```
{
  "Version" : "2012-10-17",
  "Id" : "key-test-1",
  "Statement" : [ {
    "Sid" : "Delegate to IAM policies",
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "arn:aws:iam::111122223333:root"
    },
    "Action" : "kms:*",
    "Resource" : "*"
  } ]
}
```

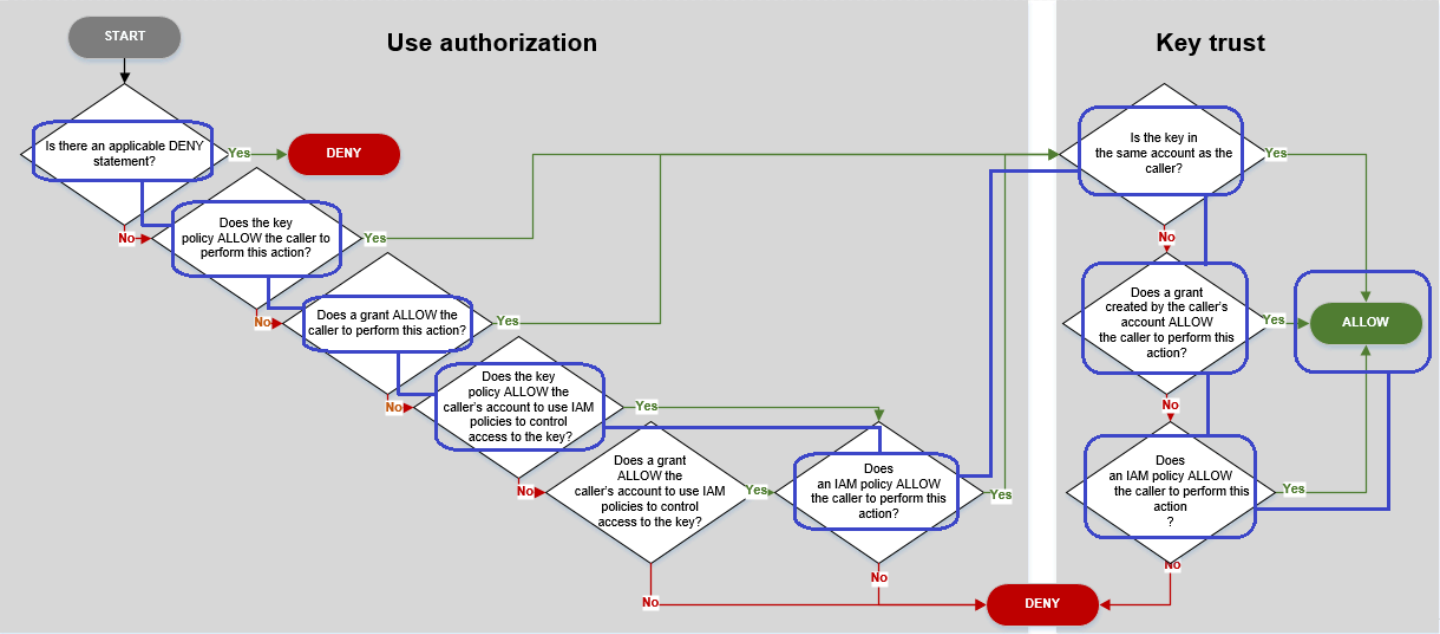
- Cependant, aucune politique de clé, aucune politique IAM ou aucun octroi ne donne à Alice l'autorisation d'utiliser la clé KMS. Par conséquent, Alice se voit refuser l'autorisation d'utiliser la clé KMS.

Exemple 2 : L'utilisateur assume un rôle autorisé à utiliser une clé KMS dans un autre Compte AWS

Bob est un utilisateur du compte 1 (111122223333). Il est autorisé à utiliser une clé KMS du compte 2 (444455556666) pour les opérations cryptographiques. Comment est-ce possible ?

> **Tip**
>
> Lors de l'évaluation des autorisations inter-comptes, n'oubliez pas que la politique de clé est spécifiée dans le compte de la clé KMS. La politique IAM est spécifiée dans le compte de l'appelant, même lorsque l'appelant se trouve dans un autre compte. Pour plus de détails sur la fourniture d'un accès inter-comptes aux clés KMS, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

- La politique de clé de la clé KMS du compte 2 autorise ce dernier à utiliser les politiques IAM pour contrôler l'accès à la clé KMS.
- La politique de clé de la clé KMS du compte 2 autorise le compte 1 à utiliser la clé KMS pour les opérations de chiffrement. Toutefois, le compte 1 doit utiliser les politiques IAM pour accorder à ses principaux l'accès à la clé KMS.
- Une politique IAM du compte 1 autorise le rôle `Engineering` à utiliser la clé KMS dans le compte 2 pour les opérations de chiffrement.
- Bob, un utilisateur du compte 1, a l'autorisation d'endosser le rôle `Engineering`.
- Enfin, Bob peut faire confiance à cette clé KMS. En effet, même si cette dernière n'appartient pas à son compte, une politique IAM de son compte lui donne l'autorisation explicite d'utiliser cette clé KMS.



Examinons les politiques qui permettent à Bob, un utilisateur du compte 1, d'utiliser la clé KMS du compte 2.

- La politique de clé de la clé KMS autorise le compte 2 (444455556666, le compte qui possède la clé KMS) à utiliser les politiques IAM pour contrôler l'accès à la clé KMS. Cette politique de clé permet également au compte 1 (111122223333) d'utiliser la clé KMS pour les opérations de chiffrement (spécifiées dans l'élément `Action` de l'instruction de politique). Toutefois, aucun utilisateur du compte 1 ne peut utiliser la clé KMS du compte 2 tant que le compte 1 n'a pas défini les politiques IAM qui accordent aux principaux l'accès à la clé KMS.

  Dans le diagramme de flux, cette politique de clé du compte 2 remplit la condition La politique de clé AUTORISE-t-elle le compte de l'appelant à utiliser les politiques IAM pour contrôler l'accès à la clé ?.

```
{
    "Id": "key-policy-acct-2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Permission to use IAM policies",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::444455556666:root"
            },
            "Action": "kms:*",
```

```
            "Resource": "*"
        },
        {
            "Sid": "Allow account 1 to use this KMS key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKey",
                "kms:GenerateDataKeyWithoutPlaintext",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
}
```

- Une politique IAM de l'appelant Compte AWS (compte 1, 111122223333) autorise le principal à effectuer des opérations cryptographiques à l'aide de la clé KMS du compte 2 (444455556666). L'élément `Action` accorde au principal les mêmes autorisations que la politique de clé du compte 2 a accordées au compte 1. Pour accorder ces autorisations au rôle `Engineering` du compte 1, cette politique en ligne est intégrée au rôle `Engineering`.

  De telles politiques IAM inter-comptes sont efficaces uniquement lorsque la politique de clé de la clé KMS du compte 2 accorde au compte 1 l'autorisation d'utiliser la clé KMS. De plus, le compte 1 peut uniquement accorder aux principaux l'autorisation d'effectuer les actions accordées au compte par la politique de clé.

  Dans le diagramme de flux, cela remplit la condition Une politique IAM autorise-t-elle l'appelant à effectuer cette action ?.

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

```
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKey",
                "kms:GenerateDataKeyWithoutPlaintext",
                "kms:DescribeKey"
            ],
            "Resource": [
                "arn:aws:kms:us-
west-2:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            ]
        }
    ]
}
```

- Le dernier élément requis est la définition du rôle `Engineering` dans le compte 1. Le document `AssumeRolePolicyDocument` dans le rôle permet à Bob d'endosser le rôle `Engineering`.

```
{
    "Role": {
        "Arn": "arn:aws:iam::111122223333:role/Engineering",
        "CreateDate": "2019-05-16T00:09:25Z",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": {
                "Principal": {
                    "AWS": "arn:aws:iam::111122223333:user/bob"
                },
                "Effect": "Allow",
                "Action": "sts:AssumeRole"
            }
        },
        "Path": "/",
        "RoleName": "Engineering",
        "RoleId": "AROA4KJY2TU23Y7NK62MV"
    }
}
```

AWS KMS glossaire du contrôle d'accès

La rubrique suivante décrit les termes et concepts importants relatifs au contrôle AWS KMS d'accès.

Authentification

L'authentification est le processus de vérification de votre identité. Pour envoyer une demande à AWS KMS, vous devez vous connecter à AWS l'aide de vos AWS informations d'identification.

Autorisation

L'autorisation donne l'autorisation d'envoyer des demandes pour créer, gérer ou utiliser AWS KMS des ressources. Par exemple, vous devez être autorisé à utiliser une clé KMS dans le cadre d'une opération de chiffrement.

Pour contrôler l'accès à vos AWS KMS ressources, utilisez des politiques clés, des politiques IAM et des subventions. Chaque clé KMS doit avoir une politique de clé. Si la politique de clé le permet, vous pouvez également utiliser des octrois et des politiques IAM pour accorder aux principaux l'accès à la clé KMS. Pour affiner votre autorisation, vous pouvez utiliser des clés de condition qui autorisent ou refusent l'accès uniquement lorsqu'une demande ou une ressource remplit les conditions que vous définissez. Vous pouvez également autoriser l'accès aux principaux auxquels vous faites confiance sur d'autres Comptes AWS.

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 des demandes vous-même, consultez AWS Signature Version 4 pour les demandes 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 plus d'informations, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center et Authentification multifactorielle AWS dans IAM 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 racine, consultez Tâches nécessitant des informations d'identification d'utilisateur racine dans le Guide de l'utilisateur IAM.

Identité fédérée

La meilleure pratique consiste à obliger les utilisateurs humains, y compris ceux qui ont besoin d'un accès administrateur, à utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide Services AWS d'informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, d'un fournisseur d'identité Web AWS Directory Service, du répertoire Identity Center ou de tout utilisateur qui y accède à l'aide des informations d'identification fournies Services AWS par le biais d'une source d'identité. Lorsque des identités fédérées y accèdent Comptes AWS, elles assument des rôles, qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous pouvez vous connecter et synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre source d'identité afin de les utiliser dans toutes vos applications Comptes AWS et applications. Pour

obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou une seule application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme telles que des mots de passe et des 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 d'effectuer une rotation des 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 nommer un groupe IAMAdminset lui donner les autorisations nécessaires pour administrer les 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 plus d'informations, consultez Cas d'utilisation pour les utilisateurs IAM 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. Pour assumer temporairement un rôle IAM dans le AWS Management Console, vous pouvez passer d'un rôle d'utilisateur à un rôle IAM (console). 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 Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

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

- Accès utilisateur fédéré : pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez 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 Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.
  - Rôle de service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de 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 EC2 instance et qui envoient des demandes AWS CLI d' AWS API. Cela est préférable au stockage des clés d'accès dans l' EC2 instance. Pour attribuer un AWS rôle à une EC2 instance et le rendre disponible pour 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 exécutés sur l' EC2 instance d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utiliser un rôle IAM pour accorder des autorisations aux applications exécutées sur des EC2 instances Amazon dans le guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur root ou session de rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur la structure et le contenu des documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui autorise l'action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client 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

Une politique AWS KMS clé est une politique basée sur les ressources qui contrôle l'accès à une clé KMS. Chaque clé KMS doit avoir une politique de clé. Vous pouvez utiliser un autre mécanisme d'autorisation pour autoriser l'accès à la clé KMS, mais uniquement si la politique de clé le permet. (Vous pouvez utiliser une politique IAM pour refuser l'accès à une clé KMS, même si la politique de clé ne l'autorise pas explicitement.)

Les politiques basées sur les ressources sont des documents de politique JSON que vous joignez à une ressource, telle qu'une clé KMS, pour contrôler l'accès à cette ressource. La politique basée sur les ressources définit les actions qu'un principal donné peut effectuer sur cette ressource et dans quelles conditions. Vous ne spécifiez pas la ressource dans une stratégie basée sur les ressources, mais vous devez spécifier un principal, tel que des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS Les politiques basées sur les ressources sont des politiques en ligne qui sont situées dans le service qui gère la ressource. Vous ne pouvez pas utiliser les politiques AWS gérées d'IAM, telles que la stratégie `AWSKeyManagementServicePowerUser` gérée, dans une stratégie basée sur les ressources.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations en résultant représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques annule 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 (SCPs) : SCPs politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans AWS Organizations. AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de votre entreprise. Si vous activez toutes les fonctionnalités d'une organisation, vous pouvez appliquer des politiques de contrôle des services (SCPs) 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 Organizations et consultez SCPs les politiques de contrôle des services dans le Guide de AWS Organizations l'utilisateur.
- Politiques de contrôle des ressources (RCPs) : RCPs politiques JSON que vous pouvez utiliser pour définir le maximum d'autorisations disponibles pour les ressources de vos comptes sans mettre à jour les politiques IAM associées à chaque ressource que vous possédez. Le RCP limite les autorisations pour les ressources des comptes membres et peut avoir un impact sur les autorisations effectives pour les identités, y compris Utilisateur racine d'un compte AWS, qu'elles appartiennent ou non à votre organisation. Pour plus d'informations sur les Organizations RCPs, y compris une liste de ces Services AWS supports RCPs, voir Politiques de contrôle des ressources (RCPs) dans le guide de AWS Organizations l'utilisateur.
- Politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de séance en résultant sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations

peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations, consultez Politiques de session dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

AWS KMS ressources

Dans AWS KMS, la ressource principale est un AWS KMS key. AWS KMS prend également en charge un alias, une ressource indépendante qui fournit un nom convivial pour une clé KMS. Certaines AWS KMS opérations vous permettent d'utiliser un alias pour identifier une clé KMS.

Chaque instance de clé KMS ou d'alias possède un Amazon Resource Name (ARN) unique avec un format standard. Dans AWS KMS les ressources, le nom du AWS service est`kms`.

- AWS KMS key

  Format de nom ARN :

  `arn:`*`AWS partition name`*`:`*`AWS service name`*`:`*`Région AWS`*`:`*`Compte AWS ID`*`:key/`*`key ID`*

  Exemple de nom ARN :

  ```
  arn:aws:kms:us-
  west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
  ```

- Alias

  Format de nom ARN :

  `arn:`*`AWS partition name`*`:`*`AWS service name`*`:`*`Région AWS`*`:`*`Compte AWS ID`*`:alias/`*`alias name`*

  Exemple de nom ARN :

```
arn:aws:kms:us-west-2:111122223333:alias/example-alias
```

AWS KMS fournit un ensemble d'opérations d'API permettant de travailler avec vos AWS KMS ressources. Pour plus d'informations sur l'identification des clés KMS dans les opérations AWS Management Console et AWS KMS API, consultezIdentifiants clés () KeyId. Pour obtenir la liste des AWS KMS opérations, consultez la référence de AWS Key Management Service l'API.

Création d'une clé KMS

Vous pouvez créer AWS KMS keys dans ou AWS Management Console en utilisant l'CreateKeyopération ou la AWS::KMS::Key AWS CloudFormation ressource. Au cours de ce processus, vous définissez la politique de clé pour la clé KMS, que vous pouvez modifier à tout moment. Vous sélectionnez également les valeurs suivantes qui définissent le type de clé KMS que vous créez. Vous ne pouvez pas modifier ces propriétés après la création de la clé KMS.

Type de clé KMS

Le type de clé est une propriété qui détermine le type de clé cryptographique créé. AWS KMS propose trois types de clés pour protéger les données :

- Clés symétriques AES (Advanced Encryption Standard)

  Clés de 256 bits utilisées dans le mode Galois Counter Mode (GCM) d'AES pour fournir des données authentifiées d'une taille inférieure à encryption/decryption 4 Ko. Il s'agit du type de clé le plus courant. Il est utilisé pour protéger les autres clés de chiffrement de données utilisées dans vos applications et crypter ainsi vos données en Services AWS votre nom.

- Clés RSA, courbe elliptique ou (régions de SM2 Chine uniquement) asymétriques

  Ces clés sont disponibles en différentes tailles et supportent de nombreux algorithmes. Ils peuvent être utilisés pour le chiffrement et le déchiffrement, la signature et la vérification, ou pour dériver des opérations de secret partagé en fonction du choix de l'algorithme.

- Clés symétriques pour effectuer des opérations de codes d'authentification de message basés sur le hachage (HMAC)

  Ces clés sont des clés de 256 bits utilisées pour les opérations de signature et de vérification.

  Les clés KMS ne peuvent pas être exportées depuis le service en texte brut. Ils sont générés par et ne peuvent être utilisés que dans les modules de sécurité matériels (HSMs) utilisés par le service. Il s'agit de la propriété de sécurité fondamentale de AWS KMS garantir que les clés ne sont pas compromises.

Utilisation des clés

L'utilisation de la clé est une propriété qui détermine les opérations cryptographiques prises en charge par la clé. Les clés KMS peuvent être utilisées comme clé `ENCRYPT_DECRYPT`ou `SIGN_VERIFY`, `GENERATE_VERIFY_MAC`, ou `KEY_AGREEMENT`. Chaque

clé KMS ne peut avoir qu'un seul type d'utilisation de clé. L'utilisation d'une clé KMS pour plusieurs types d'opérations rend le produit des deux opérations plus vulnérable aux attaques.

Spécification clé

La spécification de la clé est une propriété qui représente la configuration de chiffrement d'une clé. La signification de la spécification de la clé diffère selon le type de clé.

Pour les clés KMS, la spécification de clé détermine si la clé KMS est symétrique ou asymétrique. Elles déterminent également le type d'éléments de clé et les algorithmes pris en charge.

La spécification de clé par défaut, SYMMETRIC_DEFAULT, représente une clé de chiffrement symétrique de 256 bits. Pour une description détaillée de toutes les spécifications clés prises en charge, voirRéférence de spécification clé.

Origine du matériau clé

L'origine des éléments de clé est une propriété de clé KMS qui identifie la source des éléments de clé dans la clé KMS. Vous choisissez l'origine des éléments de clé lorsque vous créez la clé KMS, et vous ne pouvez pas la modifier. La source des éléments de clé affecte les caractéristiques de sécurité, de durabilité, de disponibilité, de latence et de débit de la clé KMS.

Chaque clé KMS inclut une référence à ses éléments de clé dans ses métadonnées. L'origine des éléments de clé des clés KMS de chiffrement symétrique peut varier. Vous pouvez utiliser le matériel clé qui AWS KMS génère, le matériel clé généré dans un magasin de clés personnalisé ou importer votre propre matériel clé.

Par défaut, chaque clé KMS possède des éléments de clé uniques. Toutefois, vous pouvez créer un ensemble de clés multi-région avec les mêmes éléments de clé.

Les clés KMS peuvent avoir l'une des valeurs d'origine matérielle clés suivantes :`AWS_KMS`, `EXTERNAL` (matériau clé importé), `AWS_CLOUDHSM` (clé KMS dans un magasin de AWS CloudHSM clés) ou `EXTERNAL_KEY_STORE` (clé KMS dans un magasin de clés externe).

Rubriques

- Autorisations de création de clés KMS
- Choix du type de clé KMS à créer
- Créer une clé KMS de chiffrement symétrique
- Créer une clé KMS asymétrique

- Créer une clé KMS HMAC
- Création de clés primaires multirégionales
- Création de répliques de clés multirégionales
- Création d'une clé KMS avec du matériel clé importé
- Création d'une clé KMS dans un magasin de AWS CloudHSM clés
- Création d'une clé KMS dans des magasins de clés externes

Autorisations de création de clés KMS

Pour créer une clé KMS dans la console ou à l'aide de APIs, vous devez disposer de l'autorisation suivante dans une politique IAM. Dans la mesure du possible, utilisez des clés de condition pour limiter les autorisations. Par exemple, vous pouvez utiliser la clé de KeySpec condition kms : dans une politique IAM pour permettre aux principaux de créer uniquement des clés de chiffrement symétriques.

Pour obtenir un exemple de politique IAM pour les entités qui créent des clés, veuillez consulter Autoriser un utilisateur à créer des clés KMS.

> **Note**
>
> Soyez prudent lorsque vous autorisez les principaux à gérer les balises et les alias. La modification d'une balise ou d'un alias permet d'accorder ou de refuser l'autorisation d'utiliser la clé gérée par le client. Pour plus de détails, veuillez consulter ABAC pour AWS KMS.

- kms : CreateKey est obligatoire.
- kms : CreateAlias est nécessaire pour créer une clé KMS dans la console où un alias est requis pour chaque nouvelle clé KMS.
- kms : TagResource est obligatoire pour ajouter des balises lors de la création de la clé KMS.
- iam : CreateServiceLinkedRole est nécessaire pour créer des clés primaires multirégionales. Pour en savoir plus, consultez Contrôler l'accès aux clés multirégionales.

L'PutKeyPolicyautorisation kms : n'est pas requise pour créer la clé KMS. L'autorisation `kms:CreateKey` inclut l'autorisation de définir la politique de clé initiale. Toutefois, vous devez ajouter cette autorisation à la politique de clé lors de la création de la clé KMS pour

vous assurer que vous pouvez contrôler l'accès à la clé KMS. L'alternative consiste à utiliser le BypassLockoutSafetyCheckparamètre, ce qui n'est pas recommandé.

Les clés KMS appartiennent au AWS compte dans lequel elles ont été créées. L'utilisateur IAM qui crée une clé KMS n'est pas considéré comme le propriétaire de la clé et il n'est pas automatiquement autorisé à utiliser ou à gérer la clé KMS qu'il a créée. Comme tout autre principal, le créateur de clé doit obtenir l'autorisation via une politique de clé, une politique IAM ou une autorisation. Toutefois, les principaux qui disposent de l'autorisation `kms:CreateKey` peuvent définir la politique de clé initiale et s'octroyer l'autorisation d'utiliser ou de gérer la clé.

Choix du type de clé KMS à créer

Le type de clé KMS que vous créez dépend en grande partie de la manière dont vous prévoyez d'utiliser la clé KMS, de vos exigences en matière de sécurité et de vos exigences en matière d'autorisation. Le type de clé et l'utilisation de la clé KMS déterminent les opérations cryptographiques que la clé peut effectuer. Chaque clé KMS n'a qu'une seule utilisation de la clé. L'utilisation d'une clé KMS pour plusieurs types d'opérations rend le produit de toutes les opérations plus vulnérable aux attaques.

Pour autoriser les principaux à créer des clés KMS uniquement pour une utilisation spécifique, utilisez la clé de KeyUsage condition kms :. Vous pouvez également utiliser la clé de condition `kms:KeyUsage` pour permettre aux principaux d'appeler des opérations d'API pour une clé KMS en fonction de son utilisation de clé. Par exemple, vous pouvez autoriser la désactivation d'une clé KMS uniquement si son utilisation de clé est SIGN_VERIFY.

Utilisez les conseils suivants pour déterminer le type de clé KMS dont vous avez besoin en fonction de votre cas d'utilisation.

Chiffrer et déchiffrer des données

Utilisez une clé KMS symétrique pour la plupart des cas d'utilisation nécessitant le chiffrement et le déchiffrement de données. L'algorithme de chiffrement symétrique qu'utilise AWS KMS est rapide, efficace et assure la confidentialité et l'authenticité des données. Il prend en charge le chiffrement authentifié avec des données authentifiées supplémentaires (AAD), définies comme un contexte de chiffrement. Ce type de clé KMS nécessite que l'expéditeur et le destinataire des données chiffrées disposent d' AWS informations d'identification valides pour appeler AWS KMS.

Si votre cas d'utilisation nécessite un chiffrement externe AWS par des utilisateurs qui ne peuvent pas appeler AWS KMS, les clés KMS asymétriques sont un bon choix. Vous pouvez distribuer la

clé publique de la clé KMS asymétrique pour permettre à ces utilisateurs de chiffrer des données. Aussi, vos applications qui ont besoin de déchiffrer ces données peuvent utiliser la clé privée de la clé KMS asymétrique dans AWS KMS.

Signer des messages et vérifier des signatures

Pour signer des messages et vérifier des signatures, vous devez utiliser une clé KMS asymétrique. Vous pouvez utiliser une clé KMS avec une spécification de clé qui représente une paire de clés RSA, une paire de clés à courbe elliptique (ECC), une paire de clés ML-DSA ou une paire de clés (régions de Chine SM2 uniquement). La spécification de clé que vous choisissez est déterminée par l'algorithme de signature que vous souhaitez utiliser. Plutôt que les algorithmes de signature RSA, nous vous recommandons d'utiliser les algorithmes de signature ECDSA pris en charge par les paires de clés ECC. Utilisez une paire de clés ML-DSA lors de la migration de clés RSA ou ECC vers des clés post-quantiques. Toutefois, vous devrez peut-être utiliser une spécification de clé et un algorithme de signature particuliers pour prendre en charge les utilisateurs qui vérifient les signatures en dehors d' AWS.

Chiffrer avec des paires de clés asymétriques

Pour chiffrer des données avec une paire de clés asymétriques, vous devez utiliser une clé KMS asymétrique avec une spécification de clé RSA ou une spécification de clé (régions de SM2 Chine uniquement). Pour chiffrer des données dans AWS KMS avec la clé publique d'une paire de clés KMS, utilisez l'opération Encrypt (Chiffrer). Vous pouvez également télécharger la clé publique et la partager avec les parties qui ont besoin de chiffrer des données en dehors de AWS KMS.

Lorsque vous téléchargez la clé publique d'une clé KMS asymétrique, vous pouvez l'utiliser à l'extérieur de AWS KMS. Mais il n'est plus soumis aux contrôles de sécurité qui protègent la clé KMS AWS KMS. Par exemple, vous ne pouvez pas utiliser de politiques ou de subventions AWS KMS clés pour contrôler l'utilisation de la clé publique. Vous ne pouvez pas non plus contrôler si la clé est utilisée uniquement pour le chiffrement et le déchiffrement à l'aide des algorithmes de chiffrement pris AWS KMS en charge. Pour plus d'informations, veuillez consulter Considérations spéciales pour le téléchargement de clés publiques.

Pour déchiffrer des données chiffrées avec la clé publique extérieure à AWS KMS, appelez l'opération Decrypt. L'`Decrypt`opération échoue si les données ont été chiffrées sous une clé publique à partir d'une clé KMS avec une clé d'utilisation de`SIGN_VERIFY`. Il échouera également s'il a été chiffré à l'aide d'un algorithme qui AWS KMS ne prend pas en charge la spécification de clé que vous avez sélectionnée. Pour plus d'informations sur les spécifications clés et les algorithmes pris en charge, consultezRéférence de spécification clé.

Pour éviter ces erreurs, toute personne utilisant une clé publique en dehors de AWS KMS doit enregistrer la configuration de la clé. La AWS KMS console et la GetPublicKeyréponse fournissent les informations que vous devez inclure lorsque vous partagez la clé publique.

Déterminez des secrets partagés

Pour obtenir des secrets partagés, utilisez une clé KMS avec une courbe elliptique recommandée par le NIST ou un élément clé (régions de SM2Chine uniquement). AWS KMS utilise le cofacteur de cryptographie à courbe elliptique Diffie-Hellman Primitive (ECDH) pour établir un accord clé entre deux pairs en dérivant un secret partagé à partir de leurs paires de clés publiques-privées sur courbe elliptique. Vous pouvez utiliser le secret partagé brut renvoyé par l'DeriveSharedSecretopération pour obtenir une clé symétrique capable de chiffrer et de déchiffrer les données envoyées entre deux parties, ou de les générer et de les vérifier. HMACs AWS KMS recommande de suivre les recommandations du NIST pour la dérivation de clés lorsque vous utilisez le secret partagé brut pour dériver une clé symétrique.

Générer et vérifier les codes HMAC

Pour générer et vérifier les codes d'authentification de message utilisant hash, utilisez une clé KMS HMAC. Lorsque vous créez une clé HMAC AWS KMS, elle AWS KMS crée et protège le contenu de votre clé et garantit que vous utilisez les algorithmes MAC appropriés pour votre clé. Les codes HMAC peuvent également être utilisés comme nombres pseudo-aléatoires et dans certains scénarios pour la signature symétrique et la création de jeton.

Les clés KMS HMAC sont des clés symétriques. Lors de la création d'une clé KMS HMAC dans la console AWS KMS , choisissez le type de clé `Symmetric`.

Utilisation avec les AWS services

Pour créer une clé KMS à utiliser avec un AWS service intégré AWS KMS, consultez la documentation du service. AWS les services qui chiffrent vos données nécessitent une clé KMS de chiffrement symétrique.

Outre ces considérations, les opérations de chiffrement sur des clés KMS dont les spécifications sont différentes sont soumises à des tarifs et à des quotas de demande différents. Pour plus d'informations sur la tarification AWS KMS , consultez AWS Key Management Service Tarification . Pour de plus amples informations sur les quotas de demande, veuillez consulter Quotas de demande.

Créer une clé KMS de chiffrement symétrique

Cette rubrique explique comment créer la clé KMS de base, une clé KMS de chiffrement symétrique pour une seule région dont le contenu clé provient de AWS KMS. Vous pouvez utiliser cette clé KMS pour protéger vos ressources dans un Service AWS.

Vous pouvez créer des clés KMS de chiffrement symétriques dans la AWS KMS console, à l'aide de l'CreateKeyAPI ou du AWS::KMS::Key AWS CloudFormation modèle.

La spécification de clé par défaut, SYMMETRIC_DEFAULT, est la spécification de clé pour les clés KMS de chiffrement symétriques. Lorsque vous sélectionnez le type de clé symétrique et l'utilisation de la clé de chiffrement et de déchiffrement dans la AWS KMS console, la spécification de la `SYMMETRIC_DEFAULT` clé est sélectionnée. Dans l'CreateKeyopération, si vous ne spécifiez aucune `KeySpec` valeur, SYMMETRIC_DEFAULT est sélectionné. Si vous n'avez pas de raison d'utiliser une spécification de clé différente, SYMMETRIC_DEFAULT est un bon choix.

Pour de plus amples informations sur les quotas qui s'appliquent aux clés KMS, veuillez consulter Quotas.

Utilisation de la AWS KMS console

Vous pouvez utiliser le AWS Management Console pour créer AWS KMS keys (clés KMS).

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez Create key.
5. Pour créer une clé KMS de chiffrement symétrique, pour Key type (Type de clé), choisissez Symmetric (Symétrique).

6. Dans Key usage (Utilisation de la clé), l'option Encrypt and decrypt (Chiffrer et déchiffrer) est sélectionnée pour vous.
7. Choisissez Suivant.
8. Saisissez un alias pour la clé KMS. Le nom d'alias ne peut pas commencer par **`aws/`**. Le **`aws/`** préfixe est réservé par Amazon Web Services pour être représenté Clés gérées par AWS dans votre compte.

   > **Note**
   >
   > L'ajout, la suppression ou la mise à jour d'un alias peut permettre d'accorder ou de refuser l'autorisation d'utiliser la KMS. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des alias pour contrôler l'accès aux clés KMS.

   Un alias est un nom d'affichage que vous pouvez utiliser pour identifier facilement la clé KMS. Nous vous conseillons de choisir un alias qui indique le type de données que vous envisagez de protéger ou l'application que vous pensez utiliser avec la clé KMS.

   Les alias sont requis lorsque vous créez une clé KMS dans la AWS Management Console. Ils sont facultatifs lorsque vous utilisez l'CreateKeyopération.
9. (Facultatif) Saisissez une description pour la clé KMS.

   Vous pouvez ajouter une description maintenant ou la mettre à jour à tout moment, sauf si l'état de la clé est `Pending Deletion` ou `Pending Replica Deletion`. Pour ajouter, modifier ou supprimer la description d'une clé gérée par le client existante, modifiez la description sur la page de détails de la clé KMS dans l'opération AWS Management Console ou utilisez l'UpdateKeyDescriptionopération.
10. (Facultatif) Saisissez une clé de balise et une valeur de balise facultative. Pour ajouter plus d'une balise à la clé KMS, sélectionnez Add tag (Ajouter une balise).

    > **Note**
    >
    > L'étiquetage ou le désétiquetage d'une clé KMS permet d'accorder ou refuser l'autorisation d'utilisation de cette clé KMS. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des balises pour contrôler l'accès aux clés KMS.

Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.

11. Choisissez Suivant.
12. Sélectionnez les utilisateurs et les rôles IAM qui peuvent administrer la clé KMS.

> **Remarques**
>
> Cette politique clé donne le contrôle Compte AWS total de cette clé KMS. Il permet aux administrateurs de compte d'utiliser des politiques IAM pour autoriser d'autres principals à gérer la clé KMS. Pour en savoir plus, consultez the section called “politique de clé par défaut”.
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
>
> La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

13. (Facultatif) Pour empêcher les utilisateurs et les rôles IAM sélectionnés de supprimer cette clé KMS, dans la section Key deletion (Suppression de la clé) en bas de la page, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).
14. Choisissez Suivant.
15. Sélectionnez les utilisateurs et les rôles IAM qui peuvent utiliser la clé dans les opérations de chiffrement.

> **Remarques**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des

rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

16. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour ce faire, dans la Comptes AWS section Autre au bas de la page, choisissez Ajouter un autre compte Compte AWS et entrez le numéro Compte AWS d'identification d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

> **Note**
>
> Pour autoriser les principaux des comptes externes à utiliser la clé KMS, les administrateurs du compte externe doivent créer des politiques IAM qui fournissent ces autorisations. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

17. Choisissez Suivant.
18. Passez en revue les principales déclarations de politique pour la clé. Pour modifier la politique clé, sélectionnez Modifier.
19. Choisissez Suivant.
20. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
21. Choisissez Finish (Terminer) pour créer la clé KMS.

Utilisation de l' AWS KMS API

Vous pouvez utiliser cette CreateKeyopération pour créer tous les AWS KMS keys types. Ces exemples utilisent le AWS Command Line Interface (AWS CLI). Pour obtenir des exemples dans plusieurs langages de programmation, veuillez consulter Utilisation `CreateKey` avec un AWS SDK ou une CLI.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans les champs `Description` ou `Tags`. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

L'opération suivante crée une clé de chiffrement symétrique dans une seule région basée sur le matériel clé généré par AWS KMS. Cette opération n'a aucun paramètre obligatoire. Vous pouvez également souhaiter utiliser le paramètre `Policy` pour spécifier une politique de clé. Vous pouvez modifier la politique clé (PutKeyPolicy) et ajouter des éléments facultatifs, tels qu'une description et des balises à tout moment. Vous pouvez également créer des clés asymétriques, des clés multi-régions, des clés avec des éléments de clé importés et des clés dans des magasins de clés personnalisés. Pour créer des clés de données pour le chiffrement côté client, utilisez l'GenerateDataKeyopération.

L'`CreateKey`opération ne vous permet pas de spécifier un alias, mais vous pouvez l'CreateAliasutiliser pour créer un alias pour votre nouvelle clé KMS.

Voici un exemple d'appel à l'opération `CreateKey` sans paramètre. Cette commande utilise toutes les valeurs par défaut. Elle crée une clé KMS de chiffrement symétrique avec les éléments de clé générés par AWS KMS.

```
$ aws kms create-key
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "KeySpec": "SYMMETRIC_DEFAULT",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "CreationDate": 1502910355.475,
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "MultiRegion": false
        "EncryptionAlgorithms": [
```

```
            "SYMMETRIC_DEFAULT"
        ],
    }
}
```

Si vous ne spécifiez pas de politique de clé pour votre nouvelle clé KMS, la politique de clé par défaut appliquée par `CreateKey` est différente de celle appliquée par la console lorsque vous utilisez cette dernière pour créer une nouvelle clé KMS.

Par exemple, cet appel à l'GetKeyPolicyopération renvoie la politique clé qui `CreateKey` s'applique. Il donne Compte AWS accès à la clé KMS et lui permet de créer des politiques AWS Identity and Access Management (IAM) pour la clé KMS. Pour plus d'informations sur les politiques IAM et les politiques de clé pour les clés KMS, veuillez consulter Accès aux clés KMS et autorisations

```
$ aws kms get-key-policy --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --policy-name
 default --output text
{
  "Version" : "2012-10-17",
  "Id" : "key-default-1",
  "Statement" : [ {
    "Sid" : "Enable IAM User Permissions",
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "arn:aws:iam::111122223333:root"
    },
    "Action" : "kms:*",
    "Resource" : "*"
  } ]
}
```

Créer une clé KMS asymétrique

Vous pouvez créer des clés KMS asymétriques dans la AWS KMS console, à l'aide de l'CreateKeyAPI ou du AWS::KMS::Key AWS CloudFormation modèle. Une clé KMS asymétrique représente une paire de clés publique et privée qui peut être utilisée pour le chiffrement, la signature ou la dérivation de secrets partagés. La clé privée reste à l'intérieur AWS KMS. Pour télécharger la clé publique afin de l'utiliser en dehors de AWS KMS, consultezTélécharger la clé publique.

Lorsque vous créez une clé KMS asymétrique, vous devez sélectionner une spécification de clé. Souvent, la spécification de clé que vous sélectionnez est déterminée par des exigences

réglementaires, de sécurité ou métier. Elle peut également être influencée par la taille des messages que vous devez chiffrer ou signer. En général, les clés de chiffrement plus longues résistent mieux aux attaques par force brute. Pour une description détaillée de toutes les spécifications clés prises en charge, voirRéférence de spécification clé.

AWS les services intégrés à ne prennent AWS KMS pas en charge les clés KMS asymétriques. Si vous souhaitez créer une clé KMS qui chiffre les données que vous stockez ou gérez dans un AWS service, créez une clé KMS de chiffrement symétrique.

Pour obtenir plus d'informations sur les autorisations nécessaires pour créer des clés KMS, consultez Autorisations de création de clés KMS.

Utilisation de la AWS KMS console

Vous pouvez utiliser le AWS Management Console pour créer des clés asymétriques AWS KMS keys (clés KMS). Chaque clé KMS asymétrique représente une paire de clés publique et privée.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez Create key.
5. Pour créer une clé KMS asymétrique, dans Key type (Type de clé), choisissez Asymmetric (Asymétrique).
6. Pour créer une clé KMS asymétrique pour le chiffrement de clé publique, dans Key usage (Utilisation de la clé), choisissez Encrypt and decrypt (Chiffrer et déchiffrer).

   Pour créer une clé KMS asymétrique pour signer les messages et vérifier les signatures, dans Utilisation des clés, choisissez Signer et vérifier.

Pour créer une clé KMS asymétrique afin de dériver des secrets partagés, dans Utilisation des clés, choisissez Key agreement.

Pour obtenir de l'aide sur le choix d'une valeur d'utilisation de clé, veuillez consulter Choix du type de clé KMS à créer.

7. Sélectionnez une spécification (Key spec (Spécifications de la clé)) pour votre clé KMS asymétrique.
8. Choisissez Suivant.
9. Saisissez un alias pour la clé KMS. Le nom d'alias ne peut pas commencer par **`aws/`**. Le préfixe **`aws/`** est réservé par Amazon Web Services pour représenter Clés gérées par AWS dans votre compte.

   Un alias est un nom convivial que vous pouvez utiliser pour identifier la clé KMS dans la console et dans certaines autres AWS KMS APIs. Nous vous conseillons de choisir un alias qui indique le type de données que vous envisagez de protéger ou l'application que vous pensez utiliser avec la clé KMS.

   Les alias sont requis lorsque vous créez une clé KMS dans la AWS Management Console. Vous ne pouvez pas spécifier d'alias lorsque vous utilisez l'CreateKeyopération, mais vous pouvez utiliser la console ou l'CreateAliasopération pour créer un alias pour une clé KMS existante. Pour plus de détails, consultez Alias dans AWS KMS.
10. (Facultatif) Saisissez une description pour la clé KMS.

    Saisissez une description qui explique le type de données que vous envisagez de protéger ou l'application que vous pensez utiliser avec la clé KMS.

    Vous pouvez ajouter une description maintenant ou la mettre à jour à tout moment, sauf si l'état de la clé est `Pending Deletion` ou `Pending Replica Deletion`. Pour ajouter, modifier ou supprimer la description d'une clé gérée par le client existante, modifiez la description sur la page de détails de la clé KMS dans l'opération AWS Management Console ou utilisez l'UpdateKeyDescriptionopération.
11. (Facultatif) Saisissez une clé de balise et une valeur de balise facultative. Pour ajouter plus d'une balise à la clé KMS, sélectionnez Add tag (Ajouter une balise).

    Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être

utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.

12. Choisissez Suivant.
13. Sélectionnez les utilisateurs et les rôles IAM qui peuvent administrer la clé KMS.

> **Remarques**
>
> Cette politique clé donne le contrôle Compte AWS total de cette clé KMS. Il permet aux administrateurs de compte d'utiliser des politiques IAM pour autoriser d'autres principals à gérer la clé KMS. Pour plus de détails, consultez the section called “politique de clé par défaut”.
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
>
> La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

14. (Facultatif) Pour empêcher les utilisateurs et les rôles IAM sélectionnés de supprimer cette clé KMS, dans la section Key deletion (Suppression de la clé) en bas de la page, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).
15. Choisissez Suivant.
16. Sélectionnez les utilisateurs et les rôles IAM qui peuvent utiliser la clé KMS pour les opérations crytographiques.

> **Remarques**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

17. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour cela, dans la section Autres Comptes AWS en bas de la page, sélectionnez Ajouter un autre Compte AWS et saisissez le numéro d'identification Compte AWS d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

> **Note**
>
> Pour autoriser les principaux des comptes externes à utiliser la clé KMS, les administrateurs du compte externe doivent créer des politiques IAM qui fournissent ces autorisations. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

18. Choisissez Suivant.
19. Passez en revue les principales déclarations de politique pour la clé. Pour apporter des modifications à la politique clé, sélectionnez Modifier.
20. Choisissez Suivant.
21. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
22. Choisissez Finish (Terminer) pour créer la clé KMS.

Utilisation de l' AWS KMS API

Vous pouvez utiliser cette CreateKeyopération pour créer une asymétrique AWS KMS key. Ces exemples utilisent l'AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Lorsque vous créez une clé KMS asymétrique, vous devez spécifier le paramètre `KeySpec`, qui détermine le type de clés que vous créez. Vous devez également spécifier la `KeyUsage` valeur ENCRYPT_DECRYPT, SIGN_VERIFY ou KEY_AGREEMENT. Vous ne pouvez pas modifier ces propriétés après la création de la clé KMS.

L'`CreateKey`opération ne vous permet pas de spécifier un alias, mais vous pouvez l'CreateAliasutiliser pour créer un alias pour votre nouvelle clé KMS.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans les champs `Description` ou `Tags`. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

Création d'une paire de clés KMS asymétriques pour le chiffrement public

L'exemple suivant utilise l'opération `CreateKey` pour créer une clé KMS asymétrique de clés RSA 4 096 bits conçues pour le chiffrement d'une clé publique.

```
$ aws kms create-key --key-spec RSA_4096 --key-usage ENCRYPT_DECRYPT
{
    "KeyMetadata": {
        "KeyState": "Enabled",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "Description": "",
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": 1569973196.214,
        "MultiRegion": false,
        "KeySpec": "RSA_4096",
        "CustomerMasterKeySpec": "RSA_4096",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "EncryptionAlgorithms": [
            "RSAES_OAEP_SHA_1",
            "RSAES_OAEP_SHA_256"
        ],
        "AWSAccountId": "111122223333",
        "Origin": "AWS_KMS",
        "Enabled": true
    }
}
```

Création d'une paire de clés KMS asymétriques pour la signature et la vérification

L'exemple de commande suivant crée une clé KMS asymétrique qui représente une paire de clés ECC utilisées pour la signature et la vérification. Vous ne pouvez pas créer une paire de clés de courbe elliptique pour le chiffrement et le déchiffrement.

```
$ aws kms create-key --key-spec ECC_NIST_P521 --key-usage SIGN_VERIFY
{
    "KeyMetadata": {
        "KeyState": "Enabled",
        "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "CreationDate": 1570824817.837,
        "Origin": "AWS_KMS",
        "SigningAlgorithms": [
            "ECDSA_SHA_512"
        ],
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "AWSAccountId": "111122223333",
        "KeySpec": "ECC_NIST_P521",
        "CustomerMasterKeySpec": "ECC_NIST_P521",
        "KeyManager": "CUSTOMER",
        "Description": "",
        "Enabled": true,
        "MultiRegion": false,
        "KeyUsage": "SIGN_VERIFY"
    }
}
```

Créez une paire de clés KMS asymétriques pour dériver des secrets partagés

L'exemple de commande suivant crée une clé KMS asymétrique qui représente une paire de clés ECDH utilisées pour dériver des secrets partagés. Vous ne pouvez pas créer une paire de clés de courbe elliptique pour le chiffrement et le déchiffrement.

```
$ aws kms create-key --key-spec ECC_NIST_P256 --key-usage KEY_AGREEMENT
{
    "KeyMetadata": {
        "AWSAccountId": "111122223333",
        "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "CreationDate": "2023-12-27T19:10:15.063000+00:00",
        "Enabled": true,
```

```
        "Description": "",
        "KeyUsage": "KEY_AGREEMENT",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "CUSTOMER",
        "CustomerMasterKeySpec": "ECC_NIST_P256",
        "KeySpec": "ECC_NIST_P256",
        "KeyAgreementAlgorithms": [
            "ECDH"
        ],
        "MultiRegion": false
    }
}
```

Créer une clé KMS HMAC

Vous pouvez créer des clés HMAC KMS dans la AWS KMS console à l'aide du CreateKeyAPI, ou en utilisant le AWS::KMS::Key AWS CloudFormation modèle.

Lorsque vous créez une clé HMAC KMS, vous devez sélectionner une spécification de clé. AWS KMS prend en charge plusieurs spécifications clés pour les clés HMAC KMS. La spécification de clé que vous sélectionnez pourrait être déterminée par des exigences réglementaires, de sécurité ou métier. En général, les clés plus longues résistent mieux aux attaques par force brute.

Pour obtenir plus d'informations sur les autorisations nécessaires pour créer des clés KMS, consultez Autorisations de création de clés KMS.

Utilisation de la AWS KMS console

Vous pouvez utiliser le AWS Management Console pour créer des clés HMAC KMS. Les clés KMS HMAC sont des clés symétriques avec une utilisation de clé de Generate and verify MAC (Générer et vérifier le MAC). Vous pouvez également créer des clés HMAC multi-région.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez Create key.
5. Pour Type de clé, choisissez Symétrique.

Les clés KMS HMAC sont symétriques. Vous utilisez la même clé pour générer et vérifier les balises HMAC.

6. Pour Key usage (Utilisation de la clé), choisissez Generate and verify MAC (Générer et vérifier le MAC).

   Générer et vérifier le MAC est la seule utilisation valide de clés KMS HMAC.

   > **Note**
   >
   > Key usage (Utilisation de la clé) est affiché pour les clés symétriques uniquement lorsque les clés KMS HMAC sont prises en charge dans votre région sélectionnée.

7. Sélectionnez une spécification (Key spec [Spécifications de clé]) pour votre clé KMS HMAC.

   La spécification de clé que vous sélectionnez peut être déterminée par des exigences réglementaires, de sécurité ou métier. En général, les clés plus longues sont plus sécurisées.

8. Pour créer une clé HMAC principale multi-région, dans Advanced options (Options avancées), choisissez Multi-Region key (Clé multi-région). Les propriétés partagées que vous définissez pour cette clé KMS, comme son type de clé et son utilisation de clé, seront partagés avec ses clés de réplica.

   Vous ne pouvez pas utiliser cette procédure pour créer une clé de réplica. Pour créer une clé HMAC de réplica multi-région, suivez les instructions de création d'une clé de réplica.

9. Choisissez Suivant.

10. Saisissez un alias pour la clé KMS. Le nom d'alias ne peut pas commencer par **`aws/`**. Le préfixe **`aws/`** est réservé par Amazon Web Services pour représenter Clés gérées par AWS dans votre compte.

    Nous vous recommandons d'utiliser un alias qui identifie la clé KMS en tant que clé HMAC, comme `HMAC/test-key`. Cela vous permettra d'identifier plus facilement vos clés HMAC dans la AWS KMS console, où vous pourrez trier et filtrer les clés par balises et alias, mais pas par spécification ou utilisation des clés.

    Les alias sont requis lorsque vous créez une clé KMS dans la AWS Management Console. Vous ne pouvez pas spécifier d'alias lorsque vous utilisez l'CreateKeyopération, mais vous pouvez utiliser la console ou l'CreateAliasopération pour créer un alias pour une clé KMS existante. Pour plus de détails, consultez Alias dans AWS KMS.

11. (Facultatif) Saisissez une description pour la clé KMS.

    Saisissez une description qui explique le type de données que vous envisagez de protéger ou l'application que vous pensez utiliser avec la clé KMS.

    Vous pouvez ajouter une description maintenant ou la mettre à jour à tout moment, sauf si l'état de la clé est `Pending Deletion` ou `Pending Replica Deletion`. Pour ajouter, modifier ou supprimer la description d'une clé gérée par le client existante, modifiez la description sur la page de détails de la clé KMS AWS Management Console dans le AWS Management Console ou utilisez l'UpdateKeyDescriptionopération.
12. (Facultatif) Saisissez une clé de balise et une valeur de balise facultative. Pour ajouter plus d'une balise à la clé KMS, sélectionnez Add tag (Ajouter une balise).

    Envisagez d'ajouter une balise qui identifie la clé en tant que clé HMAC, comme `Type=HMAC`. Cela vous permettra d'identifier plus facilement vos clés HMAC dans la AWS KMS console, où vous pourrez trier et filtrer les clés par balises et alias, mais pas par spécification ou utilisation des clés.

    Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.
13. Choisissez Suivant.
14. Sélectionnez les utilisateurs et les rôles IAM qui peuvent administrer la clé KMS.

    > **Remarques**
    >
    > Cette politique clé donne le contrôle Compte AWS total de cette clé KMS. Il permet aux administrateurs de compte d'utiliser des politiques IAM pour autoriser d'autres principals à gérer la clé KMS. Pour plus de détails, consultez the section called “politique de clé par défaut”.
    >
    > Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
    >
    > La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet

identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

15. (Facultatif) Pour empêcher les utilisateurs et les rôles IAM sélectionnés de supprimer cette clé KMS, dans la section Key deletion (Suppression de la clé) en bas de la page, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).
16. Choisissez Suivant.
17. Sélectionnez les utilisateurs et les rôles IAM qui peuvent utiliser la clé KMS pour les opérations cryptographiques.

> **Remarques**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
>
> La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

18. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour cela, dans la section Autres Comptes AWS en bas de la page, sélectionnez Ajouter un autre Compte AWS et saisissez le numéro d'identification Compte AWS d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

> **Note**
>
> Pour autoriser les principaux des comptes externes à utiliser la clé KMS, les administrateurs du compte externe doivent créer des politiques IAM qui fournissent ces autorisations. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

19. Choisissez Suivant.
20. Passez en revue les principales déclarations de politique pour trouver la clé. Pour apporter des modifications à la politique clé, sélectionnez Modifier.
21. Choisissez Suivant.
22. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
23. Choisissez Finish (Terminer) pour créer la clé KMS HMAC.

Utilisation de l' AWS KMS API

Vous pouvez utiliser cette CreateKeyopération pour créer une clé HMAC KMS. Ces exemples utilisent l'AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Lorsque vous créez une clé KMS HMAC, vous devez spécifier le paramètre `KeySpec`, qui détermine le type de clé KMS. Vous devez également spécifier une valeur `KeyUsage` de GENERATE_VERIFY_MAC, même s'il s'agit de la seule valeur d'utilisation de clé valide pour les clés HMAC. Pour créer une clé multi-région KMS HMAC, ajoutez le paramètre `MultiRegion` avec la valeur `true`. Vous ne pouvez pas modifier ces propriétés après la création de la clé KMS.

L'`CreateKey`opération ne vous permet pas de spécifier un alias, mais vous pouvez l'CreateAliasutiliser pour créer un alias pour votre nouvelle clé KMS. Nous vous recommandons d'utiliser un alias qui identifie la clé KMS en tant que clé HMAC, comme `HMAC/test-key`. Cela vous permettra d'identifier plus facilement vos clés HMAC dans la AWS KMS console, où vous pourrez trier et filtrer les clés par alias, mais pas par spécification ou utilisation des clés.

Si vous essayez de créer une clé HMAC KMS dans un fichier Région AWS dans lequel les clés HMAC ne sont pas prises en charge, l'`CreateKey`opération renvoie un `UnsupportedOperationException`

L'exemple suivant utilise l'opération `CreateKey` pour créer une clé KMS HMAC de 512 bits.

```
$ aws kms create-key --key-spec HMAC_512 --key-usage GENERATE_VERIFY_MAC
{
    "KeyMetadata": {
        "KeyState": "Enabled",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
```

```
        "Description": "",
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": 1669973196.214,
        "MultiRegion": false,
        "KeySpec": "HMAC_512",
        "CustomerMasterKeySpec": "HMAC_512",
        "KeyUsage": "GENERATE_VERIFY_MAC",
        "MacAlgorithms": [
            "HMAC_SHA_512"
        ],
        "AWSAccountId": "111122223333",
        "Origin": "AWS_KMS",
        "Enabled": true
    }
}
```

Création de clés primaires multirégionales

Vous pouvez créer une clé primaire multirégionale dans la AWS KMS console ou à l'aide de l' AWS KMS API. Vous pouvez créer la clé primaire partout Région AWS où les clés multirégionales sont prises AWS KMS en charge.

Pour créer une clé primaire multirégionale, le principal a besoin des mêmes autorisations que celles dont il a besoin pour créer n'importe quelle clé KMS, y compris l'CreateKeyautorisation kms : dans une politique IAM. Le principal a également besoin de l'CreateServiceLinkedRoleautorisation iam :. Vous pouvez utiliser la clé de MultiRegionKeyType condition kms : pour autoriser ou refuser l'autorisation de créer des clés primaires multirégionales.

> **Note**
>
> Lorsque vous créez votre clé primaire multirégionale, prenez bien en compte les utilisateurs et les rôles IAM que vous sélectionnez pour administrer et utiliser la clé. Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles l'autorisation de gérer la clé KMS.
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Utilisation de la AWS KMS console

Pour créer une clé primaire multirégionale dans la AWS KMS console, utilisez le même processus que celui que vous utiliseriez pour créer n'importe quelle clé KMS. Vous sélectionnez une clé multi-région dans Advanced options (Options avancées). Pour obtenir des instructions complètes, veuillez consulter Création d'une clé KMS.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez Create key.
5. Sélectionnez un type de clé symétrique ou asymétrique. Les clés symétriques sont les valeurs par défaut.

   Vous pouvez créer des clés symétriques et asymétriques multi-région, y compris des clés HMAC KMS multi-région, qui sont symétriques.
6. Sélectionnez l'utilisation de vos clés. Encrypt and decrypt (Chiffrer et déchiffrer) est la valeur par défaut.

   Pour obtenir de l'aide, veuillez consulter Création d'une clé KMS, the section called “Créer une clé KMS asymétrique” ou the section called “Créer une clé KMS HMAC”.
7. (Facultatif) Développez Options avancées.
8. Sous Origine du matériau clé, pour avoir AWS KMS généré le matériel clé que vos clés principales et répliques partageront, choisissez KMS. Si vous importez des éléments de clé dans les clés principales et clés de réplica, sélectionnez External (Import key material) (Externe (Importation des éléments de clé)).
9. Sous Régionalité, choisissez la clé multirégionale.

Vous ne pouvez pas modifier ce paramètre une fois que vous créez la clé KMS.

10. Saisissez un alias pour la clé principale.

    Les alias ne sont pas une propriété partagée de clés multi-région. Vous pouvez attribuer à votre clé primaire multirégionale et à ses répliques le même alias ou des alias différents. AWS KMS ne synchronise pas les alias des clés multirégionales.

    > **Note**
    >
    > L'ajout, la suppression ou la mise à jour d'un alias peut permettre d'accorder ou de refuser l'autorisation d'utiliser la KMS. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des alias pour contrôler l'accès aux clés KMS.

11. (Facultatif) Saisissez une description de la clé principale.

    Les descriptions ne sont pas une propriété partagée des clés multi-région. Vous pouvez donner à votre clé primaire multirégionale et à ses répliques la même description ou des descriptions différentes. AWS KMS ne synchronise pas les descriptions des clés multirégionales.

12. (Facultatif) Saisissez une clé de balise et une valeur de balise facultative. Pour affecter plus d'une balise à la clé principale, sélectionnez Add tag (Ajouter une balise).

    Les balises ne sont pas une propriété partagée des clés multi-région. Vous pouvez attribuer à votre clé principale multi-région et à ses réplicas les mêmes balises ou des balises différentes. AWS KMS ne synchronise pas les balises des clés multi-région. Vous pouvez modifier les balises des clés KMS à tout moment.

    > **Note**
    >
    > L'ajout ou la suppression d'une balise sur une KMS permet d'accorder ou de refuser l'autorisation d'utilisation de cette clé KMS. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des balises pour contrôler l'accès aux clés KMS.

13. Sélectionnez les utilisateurs et les rôles IAM qui peuvent administrer la clé principale.

> **Remarques**
>
> - Cette étape démarre le processus de création d'une politique de clé pour la clé principale. Les politiques de clé ne sont pas une propriété partagée des clés multi-région. Vous pouvez attribuer à votre clé primaire multirégionale et à ses répliques la même politique clé ou des politiques clés différentes. AWS KMS ne synchronise pas les politiques clés des clés multirégionales. Vous pouvez modifier la politique de clé d'une clé KMS à tout moment.
> - Lorsque vous créez une clé primaire multirégionale, pensez à utiliser la politique de clé par défaut générée par la console. Si vous modifiez cette politique, la console ne fournira pas d'étapes pour sélectionner les administrateurs et les utilisateurs clés lors de la création de répliques de clés, et elle n'ajoutera pas non plus les déclarations de politique correspondantes. Par conséquent, vous devrez les ajouter manuellement.
> - La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

14. (Facultatif) Pour empêcher les utilisateurs et les rôles IAM sélectionnés de supprimer cette clé KMS, dans la section Key deletion (Suppression de la clé) en bas de la page, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).
15. Choisissez Suivant.
16. Sélectionnez les utilisateurs et les rôles IAM qui peuvent utiliser la clé KMS pour les opérations crytographiques.

> **Remarques**
>
> La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

17. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour cela, dans la section Autres Comptes AWS en bas de la page, sélectionnez Ajouter un autre Compte AWS et saisissez le numéro d'identification Compte AWS d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

    > **Note**
    >
    > Pour autoriser les principaux des comptes externes à utiliser la clé KMS, les administrateurs du compte externe doivent créer des politiques IAM qui fournissent ces autorisations. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

18. Choisissez Suivant.
19. Passez en revue les principales déclarations de politique pour la clé. Pour apporter des modifications à la politique clé, sélectionnez Modifier.
20. Choisissez Suivant.
21. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
22. Choisissez Terminer pour créer la clé primaire multirégionale.

Utilisation de l' AWS KMS API

Pour créer une clé primaire multirégionale, utilisez l'CreateKeyopération. Utilisez le paramètre `MultiRegion` avec la valeur `True`.

Par exemple, la commande suivante crée une clé primaire multirégionale dans celle de l'appelant (Région AWS us-east-1). Elle accepte les valeurs par défaut pour toutes les autres propriétés, y compris la politique de clé. Les valeurs par défaut pour les clés principales multi-région sont les mêmes que les valeurs par défaut pour toutes les autres clés KMS, y compris la politique de clé par défaut. Cette procédure crée une clé de chiffrement symétrique, la clé KMS par défaut.

La réponse inclut l'élément `MultiRegion` et l'élément `MultiRegionConfiguration` avec des sous-éléments et des valeurs typiques pour une clé principale multi-région sans clés de réplica. L'ID de clé d'une clé multi-région commence toujours par `mrk-`.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans les champs `Description` ou `Tags`. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

```
$ aws kms create-key --multi-region
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "KeySpec": "SYMMETRIC_DEFAULT",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "CreationDate": 1606329032.475,
        "Arn": "arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "AWSAccountId": "111122223333",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "MultiRegion": true,
        "MultiRegionConfiguration": {
            "MultiRegionKeyType": "PRIMARY",
            "PrimaryKey": {
                "Arn": "arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                "Region": "us-east-1"
            },
            "ReplicaKeys": [ ]
      }
    }
}
```

Création de répliques de clés multirégionales

Vous pouvez créer une réplique de clé multirégionale dans la AWS KMS console, en utilisant l'ReplicateKeyopération ou en utilisant un AWS::KMS::ReplicaKey AWS CloudFormation modèle. Vous ne pouvez pas utiliser CreateKeycette opération pour créer une clé de réplique.

Vous pouvez utiliser ces procédures pour répliquer n'importe quelle clé principale multi-région, y compris une clé KMS de chiffrement symétrique, une clé KMS asymétrique ou une clé KMS HMAC.

Lorsque cette opération est terminée, la nouvelle clé de réplica a un état de clé de `Creating`. Cet état clé devient `Enabled` (ou `PendingImport` si vous créez une clé multirégionale avec du matériel clé importé) après quelques secondes lorsque le processus de création de la nouvelle clé de réplique est terminé. Alors que l'état de la clé est `Creating`, vous pouvez gérer la clé, mais vous ne pouvez pas encore l'utiliser dans les opérations cryptographiques. Si vous créez et utilisez la clé de réplique par programmation, réessayez `KMSInvalidStateException` ou appelez DescribeKeypour vérifier sa `KeyState` valeur avant de l'utiliser.

Si vous supprimez par erreur une clé de réplica, vous pouvez utiliser cette procédure pour la recréer. Si vous répliquez la même clé primaire dans la même région, la nouvelle clé de réplica que vous allez créer aura les mêmes propriétés partagées que la clé de réplica d'origine.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

Pour utiliser un AWS CloudFormation modèle afin de créer une réplique de clé, reportez-vous AWS::KMS::ReplicaKeyau guide de l'AWS CloudFormation utilisateur.

Étape 1 : Choisissez les régions de réplication

Vous choisissez généralement de répliquer une clé multirégionale dans un fichier en Région AWS fonction de votre modèle commercial et des exigences réglementaires. Par exemple, vous pouvez répliquer une clé dans les régions où vous conservez vos ressources. Ou, pour vous conformer à une exigence de reprise après sinistre, vous pouvez répliquer une clé dans des régions géographiquement éloignées.

Les AWS KMS exigences relatives aux régions de réplication sont les suivantes. Si la région que vous choisissez n'est pas conforme à ces exigences, les tentatives de réplication d'une clé échouent.

- Une clé multi-région associée par région — Vous ne pouvez pas créer de clé de réplica dans la même région que sa clé principale ou dans la même région qu'un autre réplica de la clé principale.

  Si vous essayez de répliquer une clé primaire dans une région qui possède déjà un réplica de cette clé, la tentative échouera. Si la clé de réplica actuelle dans la région affiche l'état de clé `PendingDeletion`, vous pouvez annuler la suppression de la clé de réplica ou attendre que la clé de réplica soit supprimée.
- Plusieurs clés multi-région non associées dans la même région — Vous pouvez avoir plusieurs clés multi-région non associées dans la même région. Vous pouvez par exemple avoir deux clés principales multi-région dans la région `us-east-1`. Chacune des clés principales peut avoir une clé de réplica dans la région `us-west-2`.
- Régions dans la même partition — La région de clé de réplica doit être dans la même partition AWS que la région de clé principale.
- La région doit être activée — Si une région est désactivée par défaut, vous ne pouvez pas créer de ressources dans cette région tant qu'elle n'est pas activée pour votre Compte AWS.

Étape 2 : Création de répliques de clés

> **Note**
>
> Lorsque vous créez des clés de réplique, prenez bien en compte les utilisateurs et les rôles IAM que vous sélectionnez pour administrer et utiliser la clé de réplique. Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles l'autorisation de gérer la clé KMS.
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

Utilisation de la AWS KMS console

Dans la AWS KMS console, vous pouvez créer une ou plusieurs répliques d'une clé primaire multirégionale au cours de la même opération.

Cette procédure est similaire à la création d'une clé KMS standard à région unique dans la console. Toutefois, comme une clé de réplica est basée sur la clé principale, vous ne sélectionnez pas de valeurs pour les propriétés partagées, telles que les spécifications (symétrique ou asymétrique), l'utilisation ou l'origine de la clé.

Vous spécifiez des propriétés qui ne sont pas partagées, notamment un alias, des balises, une description et une politique de clé. Par commodité, la console affiche les valeurs de propriété actuelles de la clé principale, mais vous pouvez les modifier. Même si vous conservez les valeurs de la clé primaire, ces valeurs AWS KMS ne sont pas synchronisées.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Sélectionnez l'ID de clé ou l'alias d'une clé principale multi-région. Cela ouvre la page des détails de clé pour la clé KMS.

   Pour identifier une clé principale multi-région, utilisez l'icône de l'outil dans le coin supérieur droit pour ajouter la colonne Regionality (Régionalité) dans la table.
5. Cliquez sur l'onglet Regionality (Régionalité).
6. Dans la section Related multi-Region keys (Clés multi-région associées), choisissez Create new replica keys (Créer de nouvelles clés de réplica).

   La section Related multi-Region keys (Clés multi-région associées) affiche la région de la clé principale et de ses clés de réplica. Vous pouvez utiliser cet affichage pour vous aider à choisir la région pour votre nouvelle clé de réplica.
7. Sélectionnez une ou plusieurs Régions AWS. Cette procédure crée une clé de réplica dans chacune des régions que vous sélectionnez.

Le menu inclut uniquement les régions situées dans la même AWS partition que la clé primaire. Les régions qui ont déjà une clé multi-région associée sont affichées, mais ne peuvent pas être sélectionnées. Vous n'êtes peut-être pas autorisé à répliquer une clé dans toutes les régions du menu.

Lorsque vous avez terminé de choisir les régions, fermez le menu. Les régions que vous avez choisies s'affichent. Pour annuler la réplication dans une région, cliquez sur le X en regard du nom de la région.

8. Saisissez un alias pour la clé de réplica.

   La console affiche l'un des alias actuels de la clé principale, mais vous pouvez le modifier. Vous pouvez attribuer à votre clé principale multi-région et à ses réplicas le même alias ou des alias différents. Les alias ne sont pas une propriété partagée des clés multirégionales. AWS KMS ne synchronise pas les alias des clés multirégionales.

   L'ajout, la suppression ou la mise à jour d'un alias peut permettre d'accorder ou de refuser l'autorisation d'utiliser la KMS. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des alias pour contrôler l'accès aux clés KMS.

9. (Facultatif) Saisissez une description de la clé de réplica.

   La console affiche la description actuelle de la clé principale, mais vous pouvez la modifier. Les descriptions ne sont pas une propriété partagée des clés multi-région. Vous pouvez donner à votre clé primaire multirégionale et à ses répliques la même description ou des descriptions différentes. AWS KMS ne synchronise pas les descriptions des clés multirégionales.

10. (Facultatif) Saisissez une clé de balise et une valeur de balise facultative. Pour affecter plus d'une balise à la clé de réplica, sélectionnez Add tag (Ajouter une balise).

    La console affiche les balises actuellement attachées à la clé principale, mais vous pouvez les modifier. Les balises ne sont pas une propriété partagée des clés multi-région. Vous pouvez attribuer à votre clé primaire multirégionale et à ses répliques les mêmes balises ou des balises différentes. AWS KMS ne synchronise pas les balises des clés multirégionales.

    L'ajout ou la suppression d'une balise sur une KMS permet d'accorder ou de refuser l'autorisation d'utilisation de cette clé KMS. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des balises pour contrôler l'accès aux clés KMS.

11. Sélectionnez les utilisateurs et les rôles IAM qui peuvent administrer la clé de réplica.

> **Remarques**
>
> - Si vous avez modifié la politique de clé par défaut lors de la création de votre clé primaire multirégionale, la console ne vous demandera pas de sélectionner des administrateurs ou des utilisateurs clés (étapes 11 à 15) lors de la création de la réplique de clés. Dans ce cas, vous devrez ajouter manuellement les autorisations nécessaires pour les administrateurs et utilisateurs clés à la politique clé en sélectionnant Modifier à l'étape Modifier la politique clé (étape 17).
> - Cette étape démarre le processus de création d'une politique de clé pour la clé de réplica. La console affiche la politique de clé actuelle de la clé principale, mais vous pouvez la modifier. Les politiques de clé ne sont pas une propriété partagée des clés multi-région. Vous pouvez attribuer à votre clé principale multi-région et à ses réplicas la même politique de clé ou des politiques de clé différentes. AWS KMS ne synchronise pas les politiques de clé. Vous pouvez modifier la politique de clé d'une clé KMS à tout moment.
> - La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

12. (Facultatif) Pour empêcher les utilisateurs et les rôles IAM sélectionnés de supprimer cette clé KMS, dans la section Key deletion (Suppression de la clé) en bas de la page, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).
13. Choisissez Suivant.
14. Sélectionnez les utilisateurs et les rôles IAM qui peuvent utiliser la clé KMS pour les opérations crytographiques.

> **Remarque**
>
> La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut

avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

15. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour cela, dans la section Autres Comptes AWS en bas de la page, sélectionnez Ajouter un autre Compte AWS et saisissez le numéro d'identification Compte AWS d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

    > **Note**
    >
    > Pour autoriser les principaux des comptes externes à utiliser la clé KMS, les administrateurs du compte externe doivent créer des politiques IAM qui fournissent ces autorisations. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

16. Choisissez Suivant.
17. Passez en revue les principales déclarations de politique pour la clé. Pour modifier la politique clé, sélectionnez Modifier.
18. Choisissez Suivant.
19. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
20. Choisissez Terminer pour créer la clé de réplique multirégionale.

Utilisation de l' AWS KMS API

Pour créer une clé de réplique multirégionale, utilisez l'ReplicateKeyopération. Vous ne pouvez pas utiliser CreateKeycette opération pour créer une clé de réplique. Cette opération crée une clé de réplica à la fois. La région que vous spécifiez doit respecter les exigences de région pour les clés de réplica.

Lorsque vous utilisez l'opération `ReplicateKey`, vous ne spécifiez aucune valeur pour les propriétés partagées des clés multi-région. Les valeurs des propriétés partagées sont copiées à partir de la clé principale et leur synchronisation est maintenue. Toutefois, vous pouvez spécifier des valeurs pour les propriétés qui ne sont pas partagées. Sinon, AWS KMS applique les valeurs par défaut standard pour les clés KMS, et non les valeurs de la clé primaire.

> **Note**
>
> Si vous ne spécifiez pas de valeurs pour le `DescriptionKeyPolicy`, ou `Tags` les paramètres, AWS KMS crée la clé de réplique avec une description sous forme de chaîne vide, la politique de clé par défaut et aucune balise.
> N'incluez pas d'informations confidentielles ou sensibles dans les champs `Description` ou `Tags`. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

Par exemple, la commande suivante crée une clé de réplica multi-région dans la région Asie-Pacifique (Sydney) (ap-southeast-2). Cette clé de réplica est modélisée sur la clé principale de la région USA Est (Virginie du Nord) (us-east-1), qui est identifiée par la valeur du paramètre `KeyId`. Cet exemple accepte les valeurs par défaut pour toutes les autres propriétés, y compris la politique de clé.

La réponse décrit la nouvelle clé de réplica. Elle inclut des champs pour les propriétés partagées, tels que `KeyId`, `KeySpec`, `KeyUsage`, et l'origine des éléments de clé (`Origin`). Elle inclut également des propriétés indépendantes de la clé principale, telles que la `Description`, la politique de clé (`ReplicaKeyPolicy`), et les balises (`ReplicaTags`).

La réponse inclut également l'ARN de clé et la région de la clé principale et toutes ses clés de réplica, y compris celle qui vient d'être créée dans la région ap-southeast-2. Dans cet exemple, l'élément `ReplicaKey` montre que cette clé principale a déjà été répliquée dans la région UE (Irlande) (eu-west-1).

```
$ aws kms replicate-key \
    --key-id arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab \
    --replica-region ap-southeast-2
{
    "ReplicaKeyMetadata": {
        "MultiRegion": true,
        "MultiRegionConfiguration": {
            "MultiRegionKeyType": "REPLICA",
            "PrimaryKey": {
                "Arn": "arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                "Region": "us-east-1"
            },
```

```
            "ReplicaKeys": [
                {
                    "Arn": "arn:aws:kms:ap-southeast-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "ap-southeast-2"
                },
                {
                    "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "eu-west-1"
                }
            ]
        },
        "AWSAccountId": "111122223333",
        "Arn": "arn:aws:kms:ap-southeast-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "CreationDate": 1607472987.918,
        "Description": "",
        "Enabled": true,
        "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "Origin": "AWS_KMS",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    },
    "ReplicaKeyPolicy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Id\" : \"key-
default-1\",...,
    "ReplicaTags": []
}
```

Création d'une clé KMS avec du matériel clé importé

Le matériel clé importé vous permet de protéger vos AWS ressources à l'aide des clés cryptographiques que vous générez. La présentation suivante explique comment importer vos éléments de clé dans AWS KMS. Pour plus de détails sur chaque étape du processus, consultez les rubriques correspondantes.

1. Créer une clé KMS sans élément de clé – L'origine doit être `EXTERNAL`. Une origine de clé `EXTERNAL` indique que la clé est conçue pour le matériel clé importé et AWS KMS empêche de générer du matériel clé pour la clé KMS. Dans une étape ultérieure, vous importerez vos propres éléments de clé dans cette clé KMS.

   Le matériel clé que vous importez doit être compatible avec les spécifications de la AWS KMS clé associée. Pour plus d'informations sur la compatibilité, consultezthe section called “Exigences relatives aux éléments de clé importés”.

2. Téléchargement de la clé publique d’encapsulation et du jeton d'importation – Une fois l'étape 1 terminée, téléchargez une clé publique d'encapsulage et un jeton d'importation. Ces articles protègent votre matériel clé lorsqu'il est importé AWS KMS.

   Au cours de cette étape, vous choisissez le type (« spécification de clé ») de la clé d'encapsulation RSA et l'algorithme d'encapsulage que vous utiliserez pour chiffrer vos données en transit vers AWS KMS. Vous pouvez choisir une spécification de clé d'encapsulage et un algorithme de clé d'encapsulage différents chaque fois que vous importez ou réimportez le même élément de clé.

3. Chiffrement des éléments de clé – Utilisez la clé publique d’encapsulage que vous avez téléchargée à l'étape 2 pour chiffrer les éléments de clé que vous avez créés sur votre propre système.

4. Importation des éléments de clé – Téléchargez les éléments de clé chiffrés que vous avez créés à l'étape 3 et le jeton d'importation que vous avez téléchargé à l'étape 2.

   À ce stade, vous pouvez définir un délai d'expiration facultatif. Lorsque le contenu clé importé expire, il est AWS KMS supprimé et la clé KMS devient inutilisable. Pour continuer à utiliser la clé KMS, vous devez réimporter les mêmes éléments de clé.

   Lorsque l'opération d'importation est terminée, l'état de la clé KMS passe de `PendingImport` à `Enabled`. Vous pouvez désormais utiliser la clé KMS dans des opérations de chiffrement.

AWS KMS enregistre une entrée dans votre AWS CloudTrail journal lorsque vous créez la clé KMS, que vous téléchargez la clé publique d'encapsulage et le jeton d'importation, et que vous importez le matériel clé. AWS KMS enregistre également une entrée lorsque vous supprimez du matériel clé importé ou lorsque vous AWS KMS supprimez du matériel clé expiré.

Autorisations d'importation des éléments de clé

Pour créer et gérer des clés KMS avec des éléments de clé importés, l'utilisateur a besoin d'une autorisation pour les opérations de ce processus. Vous pouvez fournir les autorisations `kms:GetParametersForImport`, `kms:ImportKeyMaterial` et `kms:DeleteImportedKeyMaterial` dans la politique de clé lorsque vous créez la clé KMS. Dans la AWS KMS console, ces autorisations sont ajoutées automatiquement pour les administrateurs de clés lorsque vous créez une clé avec une origine matérielle de clé externe.

Pour créer des clés KMS avec des éléments de clé importés, le principal a besoin des autorisations suivantes.

- kms : CreateKey (politique IAM)
  - Pour limiter cette autorisation aux clés KMS contenant du matériel clé importé, utilisez la condition de KeyOrigin politique kms : avec une valeur de`EXTERNAL`.

```
{
  "Sid": "CreateKMSKeysWithoutKeyMaterial",
  "Effect": "Allow",
  "Resource": "*",
  "Action": "kms:CreateKey",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "EXTERNAL"
    }
  }
}
```

- kms : GetParametersForImport (politique clé ou politique IAM)
  - Pour limiter cette autorisation aux demandes qui utilisent un algorithme d'encapsulage et une spécification de clé d'encapsulation particuliers, utilisez les conditions de WrappingKeySpec politique kms : WrappingAlgorithm et kms :.
- kms : ImportKeyMaterial (politique clé ou politique IAM)
  - Pour autoriser ou interdire le contenu clé qui expire et contrôler la date d'expiration, utilisez les conditions de ValidTo politique kms : ExpirationModel et kms :.

Pour réimporter du matériel clé importé, le principal a besoin des ImportKeyMaterial autorisations kms : GetParametersForImport et kms :.

Pour supprimer le matériel clé importé, le principal a besoin de l'DeleteImportedKeyMaterialautorisation kms :.

Par exemple, pour donner à l'exemple l'autorisation `KMSAdminRole` de gérer tous les aspects d'une clé KMS avec des éléments de clé importés, incluez une instruction de politique de clé telle que celle suivante dans la politique clé de la clé KMS.

```
{
  "Sid": "Manage KMS keys with imported key material",
  "Effect": "Allow",
  "Resource": "*",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/KMSAdminRole"
  },
  "Action": [
    "kms:GetParametersForImport",
    "kms:ImportKeyMaterial",
    "kms:DeleteImportedKeyMaterial"
  ]
}
```

Exigences relatives aux éléments de clé importés

L'élément de clé que vous importez doit être compatible avec les spécifications de clés de la clé KMS associée. Pour les paires de clés asymétriques, importez uniquement la clé privée de la paire. AWS KMS déduit la clé publique de la clé privée.

AWS KMS prend en charge les spécifications clés suivantes pour les clés KMS avec du matériel clé importé.

Spécification de clé KMS	Exigences relatives aux éléments de clés
Clés de chiffrement symétrique SYMMETRIC_DEFAULT	256 bits (32 octets) de données binaires Dans les régions de Chine, il doit s'agir de 128 bits (16 octets) de données binaires.
Clés HMAC HMAC_224	L'élément de clé HMAC doit être conforme à la norme RFC 2104.

Spécification de clé KMS	Exigences relatives aux éléments de clés
HMAC_256 HMAC_384 HMAC_512	La longueur de la clé doit être au moins la même que celle spécifiée par la spécification de la clé. La longueur maximale de la clé est de 1024 bits.
Clé privée asymétrique RSA RSA_2048 RSA_3072 RSA_4096	La clé privée asymétrique RSA que vous importez doit faire partie d'une paire de clés conforme à la norme RFC 3447. Module : 2 048 bits, 3 072 bits ou 4 096 bits Nombre de nombres premiers : 2 (les clés RSA à plusieurs nombres premiers ne sont pas prises en charge) Le matériel de clé asymétrique doit être codé en BER ou en DER au format PKCS (Public-Key Cryptography Standards) #8 conforme à la RFC 5208.

Spécification de clé KMS	Exigences relatives aux éléments de clés
Clé privée asymétrique à courbe elliptique ECC_NIST_P256 (secp256r1) ECC_NIST_P384 (secp384r1) ECC_NIST_P521 (secp521r1) ECC_SECG_P256K1 (secp256k1)	La clé privée asymétrique ECC que vous importez doit faire partie d'une paire de clés conforme à la norme RFC 5915. Courbe : NIST P-256, NIST P-384, NIST P-521 ou Secp256k1 Paramètres : courbes nommées uniquement (les clés ECC avec des paramètres explicites sont rejetées) Coordonnées des points publics : peuvent être compressées, non compressées ou projectives Le matériel de clé asymétrique doit être codé en BER ou en DER au format PKCS (Public-Key Cryptography Standards) #8 conforme à la RFC 5208.
Clé ML-DSA ML_DSA_44 ML_DSA_65 ML_DSA_87	L'importation de clés ML-DSA n'est pas prise en charge.

Spécification de clé KMS	Exigences relatives aux éléments de clés
SM2 clé privée asymétrique (régions de Chine uniquement)	La clé privée SM2 asymétrique que vous importez doit faire partie d'une paire de clés conforme à GM/T 0003. Courbe : SM2 Paramètres : courbe nommée uniquement (SM2les clés avec des paramètres explicites sont rejetées) Coordonnées des points publics : peuvent être compressées, non compressées ou projectives Le matériel de clé asymétrique doit être codé en BER ou en DER au format PKCS (Public-Key Cryptography Standards) #8 conforme à la RFC 5208.

Étape 1 : Création d'un matériau AWS KMS key sans clé

Par défaut, AWS KMS crée du matériel clé pour vous lorsque vous créez une clé KMS. Pour importer vos propres éléments de clé à la place, commencez par créer une clé KMS sans élément de clé. Procédez ensuite à l'importation. Pour créer une clé KMS sans élément clé, utilisez AWS KMS la console ou l'CreateKeyopération.

Pour créer une clé sans élément de clé, spécifiez l'origine de `EXTERNAL`. La propriété d'origine d'une clé KMS est immuable. Une fois que vous l'avez créée, vous ne pouvez pas convertir une clé KMS conçue pour le matériel clé importé en clé KMS avec du matériel clé provenant d'une autre source AWS KMS ou de toute autre source.

L'état d'une clé KMS avec une origine `EXTERNAL` et aucun élément de clé est `PendingImport`. Une clé KMS peut rester dans l'état `PendingImport` indéfiniment. Toutefois, vous ne pouvez pas utiliser une clé KMS dans l'état `PendingImport` dans le cadre des opérations cryptographiques. Lorsque vous importez l'élément de clé, l'état de la clé KMS passe à `Enabled`, et vous pouvez l'utiliser dans des opérations de chiffrement.

AWS KMS enregistre un événement dans votre AWS CloudTrail journal lorsque vous créez la clé KMS, que vous téléchargez la clé publique et le jeton d'importation, et que vous importez le contenu de la clé. AWS KMS enregistre également un CloudTrail événement lorsque vous supprimez du matériel clé importé ou lorsque vous AWS KMS supprimez du matériel clé expiré.

Rubriques

- Création d'une clé KMS sans élément de clé (console)
- Création d'une clé KMS sans matériel clé (AWS KMS API)

Création d'une clé KMS sans élément de clé (console)

Vous devez créer uniquement une clé KMS pour les éléments de clé importés une seule fois. Vous pouvez importer et réimporter le même élément de clé sur la clé KMS existante aussi souvent que vous avez besoin, mais vous ne pouvez pas importer d'élément de clé différent dans une clé KMS. Pour plus de détails, consultez Étape 2 : Téléchargement de la clé publique d'encapsulage et du jeton d'importation.

Pour rechercher des clés KMS existantes contenant des éléments de clé importés dans votre tableau Customer managed keys (Clés gérées par le client), utilisez l'icône en forme d'engrenage dans le coin supérieur droit pour afficher la colonne Origin (Origine) de la liste des clés KMS. Les clés importées ont une valeur Origine égale à Externe (Importation des éléments de clé).

Pour créer une clé KMS avec du matériel de clé importé, commencez par suivre les instructions de création d'une clé KMS du type de clé que vous préférez, à l'exception suivante.

Après avoir choisi l'utilisation de la clé, procédez comme suit :

1. (Facultatif) Développez Options avancées.
2. Dans le champ Key material origin (Origine des éléments de clé), sélectionnez External (Import key material) (Externe (Importation des éléments de clé)).
3. Cochez la case à côté de I understand the security, availability, and durability implications of using an imported key pour indiquer que vous comprenez les implications de l'utilisation d'éléments de clé importés. Pour de plus amples informations sur ces implications, veuillez consulter Suppression des éléments de clé importés.
4. Facultatif : pour créer une clé KMS multirégionale avec du matériel clé importé, sous Régionalité, sélectionnez Clé multirégionale.

5. Revenez aux instructions de base. Les étapes restantes de la procédure de base sont les mêmes pour toutes les clés KMS de ce type.

Lorsque vous choisissez Terminer, vous avez créé une clé KMS sans élément de clé et un statut (état de clé) En attente d'importation.

Cependant, au lieu de retourner au tableau des clés gérées par le client, la console affiche une page sur laquelle vous pouvez télécharger la clé publique et le jeton d'importation dont vous avez besoin pour importer l'élément de clé. Vous pouvez poursuivre l'étape de téléchargement dès maintenant ou choisir Annuler pour arrêter à ce stade. Vous pouvez revenir à cette étape de téléchargement à tout moment.

Suivant: Étape 2 : Téléchargement de la clé publique d'encapsulage et du jeton d'importation.

Création d'une clé KMS sans matériel clé (AWS KMS API)

Pour utiliser l'AWS KMS API afin de créer une clé KMS de chiffrement symétrique sans clé, envoyez une CreateKeydemande avec le `Origin` paramètre défini sur. `EXTERNAL` L'exemple suivant montre comment procéder avec l'AWS Command Line Interface (AWS CLI).

```
$ aws kms create-key --origin EXTERNAL
```

Lorsque la commande s'exécute correctement, vous obtenez une sortie similaire à ce qui suit. La AWS KMS clé `Origin` est `EXTERNAL` et elle `KeyState` est`PendingImport`.

> **Tip**
>
> Si la commande échoue, vous pouvez voir un `KMSInvalidStateException` ou un `NotFoundException`. Vous pouvez réessayer la demande.

```
{
    "KeyMetadata": {
        "Origin": "EXTERNAL",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Description": "",
        "Enabled": false,
        "MultiRegion": false,
        "KeyUsage": "ENCRYPT_DECRYPT",
```

```
        "KeyState": "PendingImport",
        "CreationDate": 1568289600.0,
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

Copiez la valeur `KeyId` dans la sortie de votre commande pour l'utiliser dans les étapes ultérieures, puis passez à l'Étape 2 : Téléchargement de la clé publique d’encapsulage et du jeton d'importation.

> **Note**
>
> Cette commande crée une clé KMS de chiffrement symétrique avec un `KeySpec` de `SYMMETRIC_DEFAULT` et `KeyUsage` de`ENCRYPT_DECRYPT`. Vous pouvez utiliser les paramètres facultatifs `--key-spec` et `--key-usage` créer une clé asymétrique ou KMS HMAC. Pour plus d'informations, consultez l'CreateKeyopération.

Étape 2 : Téléchargement de la clé publique d’encapsulage et du jeton d'importation

Après avoir créé un élément AWS KMS key sans clé, téléchargez une clé publique encapsulée et un jeton d'importation pour cette clé KMS à l'aide de la AWS KMS console ou de l'GetParametersForImportAPI. La clé publique d'encapsulage et le jeton d'importation constituent un ensemble indivisible qui doit être utilisé ensemble.

Vous utiliserez la clé publique d'encapsulage pour chiffrer votre élément de clé pour le transport. Avant de télécharger une paire de clés d'encapsulation RSA, vous sélectionnez la longueur (spécification de clé) de la paire de clés d'encapsulation RSA et l'algorithme d'encapsulation que vous utiliserez pour chiffrer le matériel clé importé en vue de son transport à l'étape 3. AWS KMS prend également en charge la spécification de la clé SM2 d'emballage (régions de Chine uniquement).

Chaque ensemble de clés publiques d'encapsulage et de jetons d'importation est valide pendant 24 heures. Si vous ne les utilisez pas pour importer les éléments de clé dans les 24 heures après les avoir téléchargés, vous devez en télécharger de nouveaux. Vous pouvez télécharger de nouvelles clés publiques d'encapsulage et importer des ensembles de jetons à tout moment. Cela vous permet de modifier la longueur de votre clé d'encapsulage RSA (« spécification de clé ») ou de remplacer un ensemble perdu.

Vous pouvez également télécharger une clé publique d'encapsulage et un ensemble de jetons d'importation pour réimporter les mêmes éléments de clé dans une clé KMS. Vous pouvez le faire pour définir ou modifier le délai d'expiration des éléments de clé ou pour restaurer des éléments de clé expirés ou supprimés. Vous devez télécharger et rechiffrer votre contenu clé chaque fois que vous l'importez dans. AWS KMS

Utilisation de la clé publique d'encapsulage

Le téléchargement inclut une clé publique qui vous est propre Compte AWS, également appelée clé publique encapsulée.

Avant d'importer le contenu clé, vous le chiffrez à l'aide de la clé d'encapsulage publique, puis vous le téléchargez sur. AWS KMS Lorsqu'il AWS KMS reçoit le contenu de votre clé chiffrée, il le déchiffre avec la clé privée correspondante, puis le chiffre à nouveau sous une clé symétrique AES, le tout dans un module de sécurité AWS KMS matériel (HSM).

Utilisation du jeton d'importation

Le téléchargement comprend un jeton d'importation avec des métadonnées qui garantissent que vos éléments de clé sont importés correctement. Lorsque vous téléchargez le contenu de votre clé chiffrée sur AWS KMS, vous devez télécharger le même jeton d'importation que celui que vous avez téléchargé à cette étape.

Sélectionnez une spécification de clé publique d'encapsulage

Pour protéger votre contenu clé lors de l'importation, vous le chiffrez à l'aide de la clé publique d'encapsulage à partir de AWS KMS laquelle vous le téléchargez et d'un algorithme d'encapsulage pris en charge. Vous sélectionnez une spécification de clé avant de télécharger votre clé publique d'encapsulage et votre jeton d'importation. Toutes les paires de clés d'encapsulage sont générées dans les modules de sécurité AWS KMS matériels (HSMs). La clé privée ne quitte jamais le HSM en texte brut.

Caractéristiques clés du RSA Wrapping

La spécification de clé de la clé publique d'encapsulage détermine la longueur des clés de la paire de clés RSA qui protège votre élément de clé pendant son transport vers AWS KMS. En général, nous recommandons d'utiliser la clé publique d'encapsulage la plus longue qui soit pratique. Nous proposons plusieurs spécifications d'encapsulation des clés publiques pour soutenir une variété HSMs de gestionnaires clés.

AWS KMS prend en charge les spécifications clés suivantes pour les clés d'encapsulation RSA utilisées pour importer du matériel clé de tous types, sauf indication contraire.

- RSA_4096 (préféré)
- RSA_3072
- RSA_2048

> **Note**
>
> La combinaison suivante n'est PAS prise en charge : l'élément de clé ECC_NIST_P521, la spécification de clé d'encapsulage publique RSA_2048 et un algorithme d'encapsulage RSAES_OAEP_SHA_*.
> Vous ne pouvez pas directement encapsuler l'élément de clé ECC_NIST_P521 avec une clé d'encapsulage publique RSA_2048. Utilisez une clé d'encapsulage plus grande ou un algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_*.

SM2 spécification clé d'emballage (régions de Chine uniquement)

AWS KMS prend en charge les spécifications clés suivantes pour les clés SM2 d'encapsulation utilisées pour importer du matériel clé asymétrique.

- SM2

Sélectionner un algorithme d'encapsulage

Pour protéger vos éléments de clé pendant l'importation, vous les chiffrez à l'aide de la clé publique d'encapsulage téléchargée et d'un algorithme d'enveloppement pris en charge.

AWS KMS prend en charge plusieurs algorithmes d'encapsulation RSA standard et un algorithme d'encapsulation hybride en deux étapes. En général, nous vous recommandons d'utiliser l'algorithme d'encapsulage le plus sécurisé qui soit compatible avec l'élément de clé importé et les spécifications

de clé d'encapsulage. Généralement, vous choisissez un algorithme pris en charge par le module de sécurité matérielle (HSM) ou le système de gestion de clés qui protège vos éléments de clé.

Le tableau suivant présente les algorithmes d'encapsulage pris en charge pour chaque type d'élément de clé et de clé KMS. Les algorithmes sont répertoriés dans l'ordre de préférence.

Éléments de clé	Algorithme et spécification d'encapsulage pris en charge
Clés de chiffrement symétrique Clé AES 256 bits SM4 Clé 128 bits (régions de Chine uniquement)	Algorithmes d'encapsulage : RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 Algorithmes d'encapsulage obsolètes : RASES_ _V1 PKCS1 **Note** Depuis le 10 octobre 2023, AWS KMS ne prend pas en charge l'algorithme d'encapsulage RSAES_ PKCS1 _V1_5. Spécifications de clés d'encapsulage : RSA_2048 RSA_3072 RSA_4096
Clé privée RSA asymétrique	Algorithmes d'encapsulage : RSA_AES_KEY_WRAP_SHA_256 RSA_AES_KEY_WRAP_SHA_1 SM2PKE (régions de Chine uniquement)

Éléments de clé	Algorithme et spécification d'encapsulage pris en charge
	Spécifications de clés d'encapsulage : RSA_2048 RSA_3072 RSA_4096 SM2 (Régions de Chine uniquement)
Clé privée asymétrique à courbe elliptique (ECC) Vous ne pouvez pas utiliser les algorithmes d'encapsulage RSAES_OAEP_SHA_* avec la spécification de clé d'encapsulage RSA_2048 pour encapsuler l’élément de clé ECC_NIST_P521.	Algorithmes d'encapsulage : RSA_AES_KEY_WRAP_SHA_256 RSA_AES_KEY_WRAP_SHA_1 RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 SM2PKE (régions de Chine uniquement) Spécifications de clés d'encapsulage : RSA_2048 RSA_3072 RSA_4096 SM2 (Régions de Chine uniquement)

Éléments de clé	Algorithme et spécification d'encapsulage pris en charge
Clé SM2 privée asymétrique (régions de Chine uniquement)	Algorithmes d'encapsulage : RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 SM2PKE (régions de Chine uniquement) Spécifications de clés d'encapsulage : RSA_2048 RSA_3072 RSA_4096 SM2 (Régions de Chine uniquement)
Clé HMAC	Algorithmes d'encapsulage : RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 Spécifications de clés d'encapsulage : RSA_2048 RSA_3072 RSA_4096

> **Note**
>
> Les algorithmes `RSA_AES_KEY_WRAP_SHA_1` d'encapsulation `RSA_AES_KEY_WRAP_SHA_256` et de compression ne sont pas pris en charge dans les régions de Chine.

- `RSA_AES_KEY_WRAP_SHA_256` – Un algorithme d'encapsulage hybride en deux étapes qui combine le chiffrement de votre élément de clé avec une clé symétrique AES que vous générez, puis le chiffrement de la clé symétrique AES avec la clé d'encapsulage publique RSA téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_256.

  Un algorithme `RSA_AES_KEY_WRAP_SHA_*` d'encapsulage est requis pour encapsuler le contenu de la clé privée RSA, sauf dans les régions de Chine, où vous devez utiliser l'algorithme `SM2PKE` d'encapsulage.

- `RSA_AES_KEY_WRAP_SHA_1` – Un algorithme d'encapsulage hybride en deux étapes qui combine le chiffrement de votre élément de clé avec une clé symétrique AES que vous générez, puis le chiffrement de la clé symétrique AES avec la clé publique d'encapsulage RSA téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_1.

  Un algorithme `RSA_AES_KEY_WRAP_SHA_*` d'encapsulage est requis pour encapsuler le contenu de la clé privée RSA, sauf dans les régions de Chine, où vous devez utiliser l'algorithme `SM2PKE` d'encapsulage.

- `RSAES_OAEP_SHA_256` : algorithme de chiffrement RSA avec fonctionnalité OAEP (Optimal Asymmetric Encryption Padding) et fonction de hachage SHA-256.
- `RSAES_OAEP_SHA_1` : algorithme de chiffrement RSA avec fonctionnalité OAEP (Optimal Asymmetric Encryption Padding) et fonction de hachage SHA-1.
- `RSAES_PKCS1_V1_5`(Obsolète ; depuis le 10 octobre 2023, il AWS KMS ne prend pas en charge l'algorithme d'encapsulage RSAES_ PKCS1 _V1_5) — L'algorithme de chiffrement RSA avec le format de remplissage défini dans la version 1.5 de PKCS #1.
- `SM2PKE`(Régions de Chine uniquement) — Algorithme de chiffrement basé sur une courbe elliptique défini par l'OSCCA en 0003.4-2012. GM/T

Rubriques

Téléchargement de la clé publique d’encapsulage et du jeton d'importation (console)

Vous pouvez utiliser la AWS KMS console pour télécharger la clé publique d'encapsulage et le jeton d'importation.

1. Si vous venez de terminer les étapes pour créer une clé KMS sans élément de clé et que vous êtes sur la page Download wrapping key and import token (Télécharger la clé d'encapsulage et le jeton d'importation), passez directement à Step 9.
2. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
3. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
4. Dans le volet de navigation, sélectionnez Clés gérées par le client.

   > **Tip**
   >
   > Vous pouvez uniquement importer un élément de clé dans une clé KMS d'une Origine Externe (Importation d'éléments de clé). Cela indique que la clé KMS a été créée sans élément de clé. Pour ajouter la colonne Origine à votre table, dans le coin supérieur droit de la page, choisissez l'icône des paramètres ( ). Activez Origine, puis choisissez Confirmer.

5. Choisissez l'alias ou l'ID de clé de la clé KMS qui est en attente d'importation.
6. Choisissez l'onglet Cryptographic configuration (Configuration de chiffrement) et affichez ses valeurs. Les onglets se trouvent sous la section General configuration (Configuration générale).

   Vous pouvez uniquement importer un élément de clé dans des clés KMS d'une Origine Externe (Importation d'éléments de clé). Pour plus d'informations sur la création de clés KMS avec des éléments de clé importés, veuillez consulter Importation de matériel clé pour les AWS KMS clés.
7. Choisissez l'onglet Key material (Éléments de clé) puis choisissez Import key material (Importation des éléments de clé).

   L'onglet Key material (Éléments de clés) apparaît uniquement pour les clés KMS dont la valeur Origin (Origine) est définie sur (Externe (Importation d'éléments de clé).
8. Pour Sélectionner la spécification de la clé d'encapsulage, choisissez la configuration de votre clé KMS. Une fois cette clé créée, vous ne pouvez pas modifier ses spécifications.
9. Pour Sélectionner l'algorithme d'encapsulage, choisissez l'option que vous allez utiliser pour chiffrer vos éléments de clé. Pour de plus amples informations sur les options, veuillez consulter Sélectionner l'algorithme d'encapsulage.

10. Choisissez Télécharger la clé publique d'encapsulage et le jeton d'importation, puis enregistrez le fichier.

    Si vous avez une option Suivant, pour poursuivre le processus immédiatement, choisissez Suivant. Pour continuer ultérieurement, choisissez Annuler.

11. Décompressez le fichier `.zip` que vous avez enregistré à l'étape précédente (`Import_Parameters_<key_id>_<timestamp>`).

    Le dossier contient les fichiers suivants :

    - Une clé publique encapsulée dans un fichier nomméWrappingPublicKey.bin.
    - Un jeton d'importation dans un fichier nommé `ImportToken.bin`.
    - Un fichier texte nommé README.txt. Ce fichier contient des informations sur la clé publique d'encapsulage, l'algorithme d'encapsulage à utiliser pour chiffrer vos éléments de clé, et la date et l'heure d'expiration de la clé publique d'encapsulage et du jeton d'importation.

12. Pour poursuivre le processus, consultez Chiffrer vos éléments de clé.

Téléchargement de la clé publique d'encapsulage et du jeton d'importation (AWS KMS API)

Pour télécharger la clé publique et le jeton d'importation, utilisez l'GetParametersForImportAPI. Spécifiez la clé KMS qui sera associée aux éléments de clé importé. Cette clé KMS doit avoir une Origin (Origine) définie à la valeur `EXTERNAL`.

> **Note**
>
> Vous ne pouvez pas importer de matériel clé pour les clés ML-DSA KMS.

Cet exemple indique l'algorithme d'encapsulage `RSA_AES_KEY_WRAP_SHA_256`, la spécification de clé publique d'encapsulage RSA_3072 et un exemple d'ID de clé. Remplacez ces exemples de valeurs par des valeurs valides pour votre téléchargement. Pour l'ID de clé, vous pouvez utiliser un ID de clé ou le nom ARN de clé, mais vous ne pouvez pas utiliser un nom d'alias ou un ARN d'alias pour cette opération.

```
$ aws kms get-parameters-for-import \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --wrapping-algorithm RSA_AES_KEY_WRAP_SHA_256 \
```

```
    --wrapping-key-spec RSA_3072
```

Lorsque la commande s'exécute correctement, vous obtenez une sortie similaire à ce qui suit :

```
{
    "ParametersValidTo": 1568290320.0,
    "PublicKey": "public key (base64 encoded)",
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "ImportToken": "import token (base64 encoded)"
}
```

Pour préparer les données pour l'étape suivante, base64 décode la clé publique, importe le jeton et enregistre les valeurs décodées dans des fichiers.

Pour décoder la clé publique et le jeton d'importation en base64 :

1. Copiez la clé publique codée en base64 (représentée *public key (base64 encoded)* dans l'exemple de sortie), collez-la dans un nouveau fichier, puis enregistrez le fichier. Donnez au fichier un nom descriptif, par exemple `PublicKey.b64`.
2. Utilisez OpenSSL pour décoder du format base64 le contenu du fichier et enregistrer les données décodées dans un nouveau fichier. L'exemple suivant décode les données dans le fichier que vous avez enregistré à l'étape précédente (`PublicKey.b64`) et enregistre le résultat dans un nouveau fichier nommé `WrappingPublicKey.bin`.

   ```
   $ openssl enc -d -base64 -A -in PublicKey.b64 -out WrappingPublicKey.bin
   ```
3. Copiez le jeton d'importation codé en base64 (représenté *import token (base64 encoded)* dans l'exemple de sortie), collez-le dans un nouveau fichier, puis enregistrez le fichier. Donnez au fichier un nom descriptif, par exemple `importtoken.b64`.
4. Utilisez OpenSSL pour décoder du format base64 le contenu du fichier et enregistrer les données décodées dans un nouveau fichier. L'exemple suivant décode les données dans le fichier que vous avez enregistré à l'étape précédente (`ImportToken.b64`) et enregistre le résultat dans un nouveau fichier nommé `ImportToken.bin`.

   ```
   $ openssl enc -d -base64 -A -in importtoken.b64 -out ImportToken.bin
   ```

Passez à Étape 3 : Chiffrement des éléments de clé.

Étape 3 : Chiffrement des éléments de clé

Après avoir téléchargé la clé publique et le jeton d'importation, chiffrez l'élément de votre clé à l'aide de la clé publique que vous avez téléchargée et de l'algorithme d'encapsulage que vous avez spécifié. Si vous devez remplacer la clé publique ou le jeton d'importation, ou modifier l'algorithme d'encapsulage, vous devez télécharger une nouvelle clé publique et un nouveau jeton d'importation. Pour plus d'informations sur les clés publiques et les algorithmes d'encapsulation AWS KMS compatibles, reportez-vous Sélectionnez une spécification de clé publique d'encapsulage aux sections etSélectionner un algorithme d'encapsulage.

La clé doit être au format binaire. Pour plus d'informations, consultez Exigences relatives aux éléments de clé importés.

> **Note**
>
> Pour les paires de clés asymétriques, chiffrez et importez uniquement la clé privée. AWS KMS dérive la clé publique de la clé privée.
> La combinaison suivante n'est PAS prise en charge : l'élément de clé ECC_NIST_P521, la spécification de clé d'encapsulage publique RSA_2048 et un algorithme d'encapsulage RSAES_OAEP_SHA_*.
> Vous ne pouvez pas directement encapsuler l'élément de clé ECC_NIST_P521 avec une clé d'encapsulage publique RSA_2048. Utilisez une clé d'encapsulage plus grande ou un algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_*.
> Les algorithmes d'encapsulage RSA_AES_KEY_WRAP_SHA_256 et RSA_AES_KEY_WRAP_SHA_1 ne sont pas pris en charge dans les régions de Chine.

En règle générale, vous chiffrez vos éléments de clé lorsque vous les exportez à partir du module de sécurité matérielle (HSM) ou du système de gestion de clés. Pour plus d'informations sur l'exportation des clés au format binaire, consultez la documentation relative au module HSM ou au système de gestion de clés. Vous pouvez également vous reporter à la section suivante qui fournit la démonstration d'une preuve de concept à l'aide d'OpenSSL.

Lorsque vous chiffrez vos éléments de clé, utilisez le même algorithme d'encapsulage que celui que vous avez spécifié lorsque vous avez téléchargé la clé publique et le jeton d'importation. Pour trouver l'algorithme d'encapsulage que vous avez spécifié, consultez le CloudTrail journal des événements de la GetParametersForImportdemande associée.

Générer un élément de clé à tester

Les commandes OpenSSL suivantes génèrent des éléments de clés pour chaque type pris en charge à des fins de test. Ces exemples sont fournis uniquement à des fins de test et de proof-of-concept démonstration. Pour les systèmes de production, utilisez une méthode plus sécurisée pour générer votre élément de clé, notamment un module de sécurité matériel ou un système de gestion de clé.

Pour convertir les clés privées des paires de clés asymétriques au format codé DER, dirigez la commande de génération de l'élément de clé vers la commande `openssl pkcs8` suivante. Le paramètre `topk8` indique à OpenSSL de prendre une clé privée en entrée et de renvoyer une clé au format PKCS #8. (Le comportement par défaut est le contraire.)

```
openssl pkcs8 -topk8 -outform der -nocrypt
```

Les commandes suivantes génèrent des éléments de clés de test pour chaque type de clé pris en charge.

- Clé de chiffrement symétrique (32 octets)

  Cette commande génère une clé symétrique de 256 bits (chaîne aléatoire de 32 octets) et l'enregistre dans le fichier `PlaintextKeyMaterial.bin`. Vous n'avez pas besoin d'encoder cet élément de clé.

  ```
  openssl rand -out PlaintextKeyMaterial.bin 32
  ```

  Dans les régions de Chine uniquement, vous devez générer une clé symétrique de 128 bits (chaîne aléatoire de 16 octets).

  ```
  openssl rand -out PlaintextKeyMaterial.bin 16
  ```

- Clés HMAC

  Cette commande génère une chaîne d'octets aléatoire de la taille spécifiée. Vous n'avez pas besoin d'encoder cet élément de clé.

  La longueur de votre clé HMAC doit correspondre à la longueur définie par la spécification de clé de la clé KMS. Par exemple, si la clé KMS est HMAC_384, vous devez importer une clé de 384 bits (48 octets).

```
openssl rand -out HMAC_224_PlaintextKey.bin 28

openssl rand -out HMAC_256_PlaintextKey.bin 32

openssl rand -out HMAC_384_PlaintextKey.bin 48

openssl rand -out HMAC_512_PlaintextKey.bin 64
```

- Clés privées RSA

```
openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:2048 | openssl pkcs8 -topk8 -
outform der -nocrypt > RSA_2048_PrivateKey.der

openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:3072 | openssl pkcs8 -topk8 -
outform der -nocrypt > RSA_3072_PrivateKey.der

openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 | openssl pkcs8 -topk8 -
outform der -nocrypt > RSA_4096_PrivateKey.der
```

- Clés privées ECC

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:P-256 | openssl pkcs8 -topk8
 -outform der -nocrypt > ECC_NIST_P256_PrivateKey.der

openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:P-384 | openssl pkcs8 -topk8
 -outform der -nocrypt > ECC_NIST_P384_PrivateKey.der

openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:P-521 | openssl pkcs8 -topk8
 -outform der -nocrypt > ECC_NIST_P521_PrivateKey.der

openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:secp256k1 | openssl pkcs8 -
topk8 -outform der -nocrypt > ECC_SECG_P256K1_PrivateKey.der
```

- SM2 clés privées (régions de Chine uniquement)

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:sm2 | openssl pkcs8 -topk8 -
outform der -nocrypt > SM2_PrivateKey.der
```

Exemple de chiffrement des éléments de clé avec OpenSSL

Les exemples suivants montrent comment utiliser OpenSSL pour chiffrer votre élément de clé avec la clé publique que vous avez téléchargée. Pour chiffrer votre contenu clé à l'aide d'une clé SM2 publique (régions de Chine uniquement), utilisez la `SM2OfflineOperationHelper`classe. Pour plus d'informations sur les principaux types de matériaux pris en charge par chaque algorithme d'emballage, consultezthe section called “Sélectionner un algorithme d'encapsulage”.

> **Important**
>
> Ces exemples sont la démonstration d'une preuve de concept uniquement. Pour les systèmes de production, utilisez une méthode plus sécurisée (par exemple, un système de gestion de clés ou un module HSM commercial) pour générer et stocker vos éléments de clé.
> La combinaison suivante n'est PAS prise en charge : l’élément de clé ECC_NIST_P521, la spécification de clé d'encapsulage publique RSA_2048 et un algorithme d'encapsulage RSAES_OAEP_SHA_*.
> Vous ne pouvez pas directement encapsuler l’élément de clé ECC_NIST_P521 avec une clé d'encapsulage publique RSA_2048. Utilisez une clé d'encapsulage plus grande ou un algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_*.

RSAES_OAEP_SHA_1

AWS KMS prend en charge le RSAES_OAEP_SHA_1 pour les clés de chiffrement symétriques (SYMMETRIC_DEFAULT), les clés privées à courbe elliptique (ECC), les clés privées et les clés HMAC. SM2

RSAES_OAEP_SHA_1 n'est pas pris en charge pour les clés privées RSA. Vous ne pouvez pas non plus utiliser une clé publique d’encapsulage RSA_2048 avec un algorithme d'encapsulage RSAES_OAEP_SHA_* pour encapsuler une clé privée ECC_NIST_P521 (secp521r1).
Vous devez utiliser une clé d'encapsulage plus grande ou un algorithme d'encapsulage RSA_AES_KEY_WRAP.

L'exemple suivant chiffre votre élément de clé avec la clé publique que vous avez téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_1, puis l'enregistre dans le fichier `EncryptedKeyMaterial.bin`.

Dans cet exemple :

- *WrappingPublicKey.bin* est le fichier qui contient la clé publique d'encapsulage téléchargée.
- *PlaintextKeyMaterial.bin* est le fichier qui contient l'élément de clé que vous chiffres, tel que `PlaintextKeyMaterial.bin`, `HMAC_384_PlaintextKey.bin` ou `ECC_NIST_P521_PrivateKey.der`.

```
$ openssl pkeyutl \
    -encrypt \
    -in PlaintextKeyMaterial.bin \
    -out EncryptedKeyMaterial.bin \
    -inkey WrappingPublicKey.bin \
    -keyform DER \
    -pubin \
    -pkeyopt rsa_padding_mode:oaep \
    -pkeyopt rsa_oaep_md:sha1
```

RSAES_OAEP_SHA_256

AWS KMS prend en charge le RSAES_OAEP_SHA_256 pour les clés de chiffrement symétriques (SYMMETRIC_DEFAULT), les clés privées à courbe elliptique (ECC), les clés privées et les clés HMAC. SM2

RSAES_OAEP_SHA_256 n'est pas pris en charge pour les clés privées RSA. Vous ne pouvez pas non plus utiliser une clé publique d'encapsulage RSA_2048 avec un algorithme d'encapsulage RSAES_OAEP_SHA_* pour encapsuler une clé privée ECC_NIST_P521 (secp521r1). Vous devez utiliser une clé d'encapsulage plus grande ou un algorithme d'encapsulage RSA_AES_KEY_WRAP.

L'exemple suivant chiffre votre élément de clé avec la clé publique que vous avez téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_256, puis l'enregistre dans le fichier `EncryptedKeyMaterial.bin`.

Dans cet exemple :

- *WrappingPublicKey.bin* est le fichier qui contient la clé publique d'encapsulage téléchargée. Si vous avez téléchargé la clé publique à partir de la console, ce fichier est nommé `wrappingKey_`*KMS key_key_ID_timestamp* (par exemple, `wrappingKey_f44c4e20-f83c-48f4-adc6-a1ef38829760_0809092909`).

- *PlaintextKeyMaterial.bin* est le fichier qui contient l'élément de clé que vous chiffres, tel que `PlaintextKeyMaterial.bin`, `HMAC_384_PlaintextKey.bin` ou `ECC_NIST_P521_PrivateKey.der`.

```
$ openssl pkeyutl \
    -encrypt \
    -in PlaintextKeyMaterial.bin \
    -out EncryptedKeyMaterial.bin \
    -inkey WrappingPublicKey.bin \
    -keyform DER \
    -pubin \
    -pkeyopt rsa_padding_mode:oaep \
    -pkeyopt rsa_oaep_md:sha256 \
    -pkeyopt rsa_mgf1_md:sha256
```

RSA_AES_KEY_WRAP_SHA_1

L'algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_1 implique deux opérations de chiffrement.

1. Chiffrez votre élément de clé à l'aide d'une clé symétrique AES que vous générez et d'un algorithme de chiffrement symétrique AES.
2. Chiffrez la clé symétrique AES que vous avez utilisée avec la clé publique que vous avez téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_1.

L'algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_1 nécessite la version 3.x ou version ultérieure d'OpenSSL.

1. Générez une clé de chiffrement symétrique AES 256 bits

   Cette commande génère une clé de chiffrement symétrique AES composée de 256 bits aléatoires et l'enregistre dans le fichier `aes-key.bin`

   ```
   # Generate a 32-byte AES symmetric encryption key
   $ openssl rand -out aes-key.bin 32
   ```

2. Chiffrez votre élément de clé avec la clé de chiffrement symétrique AES

   Cette commande chiffre l'élément de votre clé avec la clé de chiffrement symétrique AES et enregistre le contenu de la clé chiffrée dans le fichier `key-material-wrapped.bin`.

   Dans cet exemple de commande :

   - *`PlaintextKeyMaterial.bin`* est le fichier qui contient l'élément de clé que vous importez, tel que `PlaintextKeyMaterial.bin`, `HMAC_384_PlaintextKey.bin`, `RSA_3072_PrivateKey.der`, ou `ECC_NIST_P521_PrivateKey.der`.
   - *`aes-key.bin`* est le fichier qui contient la clé de chiffrement symétrique AES 256 bits que vous avez générée dans la commande précédente.

   ```
   # Encrypt your key material with the AES symmetric encryption key
   $ openssl enc -id-aes256-wrap-pad \
           -K "$(xxd -p < aes-key.bin | tr -d '\n')" \
           -iv A65959A6 \
           -in PlaintextKeyMaterial.bin\
           -out key-material-wrapped.bin
   ```

3. Chiffrez votre clé de chiffrement symétrique AES avec la clé publique

   Cette commande chiffre votre clé de chiffrement symétrique AES avec la clé publique que vous avez téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_1, la code DER et l'enregistre dans le fichier `aes-key-wrapped.bin`.

   Dans cet exemple de commande :

   - *`WrappingPublicKey.bin`* est le fichier qui contient la clé publique d'encapsulage téléchargée. Si vous avez téléchargé la clé publique à partir de la console, ce fichier est nommé `wrappingKey_`*`KMS key_key_ID_timestamp`* (par exemple, `wrappingKey_f44c4e20-f83c-48f4-adc6-a1ef38829760_0809092909`)
   - *`aes-key.bin`* est le fichier qui contient la clé de chiffrement symétrique AES 256 bits que vous avez générée dans la première commande de cette séquence d'exemple.

   ```
   # Encrypt your AES symmetric encryption key with the downloaded public key
   $ openssl pkeyutl \
       -encrypt \
   ```

```
    -in aes-key.bin \
    -out aes-key-wrapped.bin \
    -inkey WrappingPublicKey.bin \
    -keyform DER \
    -pubin \
    -pkeyopt rsa_padding_mode:oaep \
    -pkeyopt rsa_oaep_md:sha1 \
    -pkeyopt rsa_mgf1_md:sha1
```

4. Générez le fichier à importer

   Concaténez le fichier contenant l'élément de clé chiffré et le fichier contenant la clé AES chiffrée. Enregistrez-les dans le fichier `EncryptedKeyMaterial.bin`, qui est le fichier que vous allez importer dans le Étape 4 : Importation des éléments de clé.

   Dans cet exemple de commande :

   - *key-material-wrapped.bin* est le fichier qui contient votre élément de clé chiffré.
   - *aes-key-wrapped.bin* est le fichier qui contient la clé de chiffrement AES chiffrée.

```
# Combine the encrypted AES key and encrypted key material in a file
$ cat aes-key-wrapped.bin key-material-wrapped.bin > EncryptedKeyMaterial.bin
```

RSA_AES_KEY_WRAP_SHA_256

L'algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_256 implique deux opérations de chiffrement.

1. Chiffrez votre élément de clé à l'aide d'une clé symétrique AES que vous générez et d'un algorithme de chiffrement symétrique AES.
2. Chiffrez la clé symétrique AES que vous avez utilisée avec la clé publique que vous avez téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_256.

L'algorithme d'encapsulage RSA_AES_KEY_WRAP_SHA_256 nécessite la version 3.x ou version ultérieure d'OpenSSL.

1. Générez une clé de chiffrement symétrique AES 256 bits

   Cette commande génère une clé de chiffrement symétrique AES composée de 256 bits aléatoires et l'enregistre dans le fichier `aes-key.bin`

   ```
   # Generate a 32-byte AES symmetric encryption key
   $ openssl rand -out aes-key.bin 32
   ```

2. Chiffrez votre élément de clé avec la clé de chiffrement symétrique AES

   Cette commande chiffre l'élément de votre clé avec la clé de chiffrement symétrique AES et enregistre le contenu de la clé chiffrée dans le fichier `key-material-wrapped.bin`.

   Dans cet exemple de commande :

   - *`PlaintextKeyMaterial.bin`* est le fichier qui contient l'élément de clé que vous importez, tel que `PlaintextKeyMaterial.bin`, `HMAC_384_PlaintextKey.bin`, `RSA_3072_PrivateKey.der`, ou `ECC_NIST_P521_PrivateKey.der`.
   - *`aes-key.bin`* est le fichier qui contient la clé de chiffrement symétrique AES 256 bits que vous avez générée dans la commande précédente.

   ```
   # Encrypt your key material with the AES symmetric encryption key
   $ openssl enc -id-aes256-wrap-pad \
           -K "$(xxd -p < aes-key.bin | tr -d '\n')" \
           -iv A65959A6 \
           -in PlaintextKeyMaterial.bin\
           -out key-material-wrapped.bin
   ```

3. Chiffrez votre clé de chiffrement symétrique AES avec la clé publique

   Cette commande chiffre votre clé de chiffrement symétrique AES avec la clé publique que vous avez téléchargée et l'algorithme d'encapsulage RSAES_OAEP_SHA_256, la code DER et l'enregistre dans le fichier `aes-key-wrapped.bin`.

   Dans cet exemple de commande :

   - *`WrappingPublicKey.bin`* est le fichier qui contient la clé publique d'encapsulage téléchargée. Si vous avez téléchargé la clé publique à partir de la console, ce fichier est nommé `wrappingKey_`*`KMS key_key_ID_timestamp`* (par exemple, `wrappingKey_f44c4e20-f83c-48f4-adc6-a1ef38829760_0809092909`)

- *aes-key.bin* est le fichier qui contient la clé de chiffrement symétrique AES 256 bits que vous avez générée dans la première commande de cette séquence d'exemple.

```
# Encrypt your AES symmetric encryption key with the downloaded public key
$ openssl pkeyutl \
    -encrypt \
    -in aes-key.bin \
    -out aes-key-wrapped.bin \
    -inkey WrappingPublicKey.bin \
    -keyform DER \
    -pubin \
    -pkeyopt rsa_padding_mode:oaep \
    -pkeyopt rsa_oaep_md:sha256 \
    -pkeyopt rsa_mgf1_md:sha256
```

4. Générez le fichier à importer

   Concaténez le fichier contenant l'élément de clé chiffré et le fichier contenant la clé AES chiffrée. Enregistrez-les dans le fichier `EncryptedKeyMaterial.bin`, qui est le fichier que vous allez importer dans le Étape 4 : Importation des éléments de clé.

   Dans cet exemple de commande :

   - *key-material-wrapped.bin* est le fichier qui contient votre élément de clé chiffré.
   - *aes-key-wrapped.bin* est le fichier qui contient la clé de chiffrement AES chiffrée.

```
# Combine the encrypted AES key and encrypted key material in a file
$ cat aes-key-wrapped.bin key-material-wrapped.bin > EncryptedKeyMaterial.bin
```

Passez à Étape 4 : Importation des éléments de clé.

Étape 4 : Importation des éléments de clé

Après avoir chiffré vos éléments de clé, vous pouvez importer les éléments de clé à utiliser avec une AWS KMS key. Pour importer les éléments de clé, vous devez télécharger les éléments de clé chiffrés à partir de l'Étape 3 : Chiffrement des éléments de clé et le jeton d'importation que vous avez téléchargé à l'Étape 2 : Téléchargement de la clé publique d'encapsulage et du jeton d'importation. Vous devez importer les éléments de clé dans la même clé KMS que vous avez spécifiée lorsque

vous avez téléchargé la clé publique et le jeton d'importation. Lorsque l'élément de clé est importé avec succès, l'état de la clé KMS passe à `Enabled`, et vous pouvez utiliser la clé KMS dans des opérations de chiffrement.

Lorsque vous importez les éléments de clé, vous pouvez éventuellement définir une date d'expiration pour ceux-ci. Lorsque les éléments de clé expirent, AWS KMS supprime les éléments de clé et la clé KMS devient inutilisable. Après avoir importé vos éléments de clé, vous ne pouvez pas définir, modifier ou annuler la date d'expiration de l'importation en cours. Pour modifier ces valeurs, vous devez réimporter le même élément clé.

Pour toutes les clés KMS ayant une `EXTERNAL` origine, le premier élément clé importé devient actuel et y est associé de façon permanente. Les clés de chiffrement symétriques à région unique avec `EXTERNAL` origine prennent en charge la rotation à la demande. Vous pouvez associer plusieurs matériaux clés à des clés importées qui prennent en charge la rotation à la demande. Vous devez définir le `importType` paramètre sur `NEW_KEY_MATERIAL` avec l'ImportKeyMaterialaction permettant d'associer un nouveau matériau clé à une clé KMS. Ce matériau clé n'est pas associé de façon permanente à la clé tant que vous n'avez pas effectué l'RotateKeyOnDemandaction. D'ici là, ce matériau clé est en bon `PENDING_ROTATION` état. La valeur par défaut du `ImportType` paramètre facultatif est`EXISTING_KEY_MATERIAL`. Lorsque vous omettez le `ImportType` paramètre ou que vous le spécifiez comme tel`EXISTING_KEY_MATERIAL`, vous devez importer un élément clé précédemment associé à la clé KMS.

Pour les clés asymétriques, HMAC ou KMS multirégionales avec `EXTERNAL` origine, un seul matériau clé peut être associé à la clé. AWS KMS rejettera les demandes d' ImportKeyMaterialAPI avec le `ImportType` paramètre.

Lorsque tous les éléments clés associés de façon permanente à une clé KMS sont importés, la clé KMS peut être utilisée dans des opérations cryptographiques. Si l'un de ces éléments clés est supprimé ou autorisé à expirer, l'état de la clé KMS devient inutilisable pour les opérations cryptographiques. `PendingImport`

Pour importer du matériel clé, vous pouvez utiliser la AWS KMS console ou l'ImportKeyMaterialAPI. Vous pouvez utiliser l'API directement en effectuant des requêtes HTTP ou en utilisant un AWS SDKsAWS Command Line Interfaceou Outils AWS pour PowerShell.

Lorsque vous importez le matériel clé, une ImportKeyMaterialentrée est ajoutée à votre AWS CloudTrail journal pour enregistrer l'`ImportKeyMaterial`opération. L' CloudTrail entrée est la même que vous utilisiez la AWS KMS console ou l' AWS KMS API.

Définir un délai d'expiration (facultatif)

Lorsque vous importez les éléments de clé de votre clé KMS, vous pouvez définir une date et une heure d'expiration facultatives pour les éléments de clé pouvant aller jusqu'à 365 jours à compter de la date d'importation. Lorsque le matériel clé importé expire, il est AWS KMS supprimé. Cette action change l'état de la clé KMS en `PendingImport`, ce qui l'empêche d'être utilisée dans toute opération cryptographique. Pour utiliser la clé KMS, vous devez réimporter une copie des éléments de clé originaux.

S'assurer que les éléments de clé importés expirent fréquemment peut vous aider à satisfaire aux exigences réglementaires, mais cela introduit un risque supplémentaire pour les données chiffrées au moyen de la clé KMS. Tant que vous n'avez pas réimporté une copie des éléments de clé originaux, une clé KMS dont les éléments de clé ont expiré est inutilisable, et toutes les données chiffrées au moyen de la clé KMS sont inaccessibles. Si vous ne parvenez pas à réimporter les éléments de clé pour quelque raison que ce soit, y compris la perte de votre copie des éléments de clé originaux, la clé KMS est définitivement inutilisable et les données chiffrées au moyen de la clé KMS sont irrécupérables.

Pour atténuer ce risque, assurez-vous que votre copie du matériel clé importé est accessible et concevez un système permettant de supprimer et de réimporter le matériel clé avant qu'il n'expire et n'interrompe votre AWS charge de travail. Nous vous recommandons de programmer une alerte pour l'expiration de vos éléments de clé importés, afin de disposer de suffisamment de temps pour réimporter les éléments de clé avant leur expiration. Vous pouvez également utiliser vos CloudTrail journaux pour auditer les opérations d'importation (et de réimportation) de matériel clé et de suppression de matériel clé importé, ainsi que l' AWS KMS opération de suppression de matériel clé expiré.

AWS KMS ne peut pas restaurer, récupérer ou reproduire le contenu clé supprimé. Au lieu de définir une date d'expiration, vous pouvez supprimer et réimporter périodiquement et par programmation les éléments de clé importés, mais les exigences relatives à la conservation d'une copie des éléments de clé originaux sont les mêmes.

Vous déterminez si et quand les éléments de clé importés expirent lorsque vous importez les éléments de clé importés. Vous pouvez toutefois activer ou désactiver l'expiration, ou définir une nouvelle date d'expiration en réimportant le matériel clé. Utilisez le `ExpirationModel` paramètre de ImportKeyMaterialpour activer et désactiver l'expiration (`KEY_MATERIAL_DOES_NOT_EXPIRE`) et le `ValidTo` paramètre pour définir le délai d'expiration. `KEY_MATERIAL_EXPIRES` Le délai maximal est de 365 jours à compter de la date d'importation ; il n'y a pas de minimum, mais le délai doit être dans le futur.

Description du matériau clé du set

Les clés de chiffrement symétriques à région unique ayant une `EXTERNAL` origine peuvent être associées à plusieurs éléments clés. Vous pouvez spécifier une description facultative du matériau clé lorsque vous importez du matériel clé dans de telles clés. La description peut être utilisée pour garder une trace de l'endroit où le matériau clé correspondant est conservé durablement à l'extérieur AWS KMS.

Réimportez le matériel clé

Si vous gérez une clé KMS avec des éléments de clé importés, vous pouvez avoir besoin de les réimporter. Vous pouvez réimporter des éléments de clé pour remplacer des éléments de clé expirés ou supprimés, ou pour modifier le modèle ou la date d'expiration de ceux-ci.

Vous pouvez réimporter des éléments de clé à tout moment, selon une planification qui répond à vos exigences de sécurité. Vous n'avez pas besoin d'attendre que les éléments de clé arrivent à leur date d'expiration ou en soient proches.

La procédure de réimportation du matériel clé est la même que celle que vous utilisez pour importer le matériel clé la première fois, avec les exceptions suivantes.

- Utilisez une clé KMS existante au lieu de créer une nouvelle clé KMS. Vous pouvez ignorer l'étape 1 de la procédure d'importation.
- Lorsque vous réimportez des éléments de clé, vous pouvez modifier le modèle et la date d'expiration.

Chaque fois que vous importez des éléments de clé pour une clé KMS, vous devez télécharger et utiliser une nouvelle clé d'encapsulage et un nouveau jeton d'importation pour la clé KMS. La procédure d'encapsulage n'affecte pas le contenu de l’élément de clé. Vous pouvez ainsi utiliser différentes clés d'encapsulage et algorithmes d’encapsulation différents pour importer le même élément de clé.

Importer de nouveaux documents clés

Pour effectuer une rotation à la demande sur une clé KMS de chiffrement symétrique avec du matériel de clé importé, vous devez d'abord importer un nouveau contenu de clé, non associé auparavant à la clé. Utilisez l'ImportKeyMaterialopération avec le `ImportType` paramètre défini sur `NEW_KEY_MATERIAL` pour accomplir cette tâche. Le matériau clé importé de cette manière sera

conservé jusqu'à ce que `PENDING_ROTATION` vous effectuiez l'RotateKeyOnDemandopération ou que vous fassiez pivoter la clé dans le AWS Management Console. Une clé KMS peut contenir au plus un élément `PENDING_ROTATION` clé à tout moment.

Importer les éléments de clé (console)

Vous pouvez utiliser le AWS Management Console pour importer du matériel clé.

1. Si vous êtes sur la page Téléchargez votre élément de clé encapsulé, passez à Step 8.
2. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
3. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
4. Dans le volet de navigation, sélectionnez Clés gérées par le client.
5. Choisissez l'ID de clé ou l'alias de la clé KMS pour laquelle vous avez téléchargé la clé publique et le jeton d'importation.
6. Choisissez l'onglet Cryptographic configuration (Configuration de chiffrement) et affichez ses valeurs. Les onglets se trouvent sur la page détaillée d'une clé KMS située sous la section General configuration (Configuration générale).

   Vous pouvez uniquement importer un élément de clé dans des clés KMS d’une Origine Externe (Importation d'éléments de clé). Pour plus d'informations sur la création de clés KMS avec des éléments de clé importés, veuillez consulter Importation de matériel clé pour les AWS KMS clés.
7. Pour les clés asymétriques, HMAC et multirégionales, choisissez l'onglet Matériau clé, puis sélectionnez Importer le matériau clé. Pour les clés de chiffrement symétriques à région unique, choisissez l'onglet Matériau des clés et rotations. Choisissez ensuite Importer le matériel clé initial, Importer un nouveau matériau clé ou Réimporter le matériel clé. L'option Réimporter les matériaux clés est disponible dans le `Actions` menu du tableau des matériaux clés.

   Si vous avez téléchargé l’élément de clé, le jeton d’importation et chiffré l’élément de clé, choisissez Next (Suivant).
8. Dans la section Éléments clés chiffrés et jeton d'importation, procédez comme suit.

   a. Sous Élément de clé encapsulé, choisissez Choisir un fichier. Puis, chargez le fichier qui inclut vos clés encapsulées (chiffrées).

   b. Sous Jeton d'importation, choisissez Charger un fichier. Chargez le fichier qui inclut le jeton d'importation que vous avez téléchargé.

9. Sous Expiration option (Option d'expiration), déterminez si l'élément de clé expire. Pour définir la date et l'heure d'expiration, choisissez Date d'expiration des clés, et utilisez le calendrier pour sélectionner une date et une heure. Vous pouvez spécifier une date jusqu'à 365 jours à compter de la date et de l'heure actuelles.
10. Pour les clés de chiffrement symétriques, vous pouvez éventuellement spécifier une description du contenu clé importé.
11. Choisissez Importer le matériel clé.

Importer du matériel clé (AWS KMS API)

Pour importer du matériel clé, utilisez l'ImportKeyMaterialopération. Les exemples suivants utilisent l'AWS CLI, mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Pour utiliser cet exemple :

1. Remplacer *1234abcd-12ab-34cd-56ef-1234567890ab* par l'ID de la clé KMS que vous avez spécifié lorsque vous avez téléchargé la clé publique et le jeton d'importation. Pour identifier la clé KMS, utilisez son ID de clé ou son ARN de clé. Vous ne pouvez pas utiliser un nom d'alias ou un ARN d'alias pour cette opération.
2. Remplacez *EncryptedKeyMaterial.bin* par le nom du fichier qui contient les clés chiffrées.
3. Remplacez *ImportToken.bin* par le nom du fichier qui contient le jeton d'importation.
4. Si vous souhaitez que l'élément de clé importé expire, définissez la valeur du paramètre `expiration-model` sur sa valeur par défaut, `KEY_MATERIAL_EXPIRES`, ou omettez le paramètre `expiration-model`. Procédez ensuite au remplacement de la valeur du paramètre `valid-to` par la date et l'heure auxquelles vous souhaitez que l'élément de clé expire. La date et l'heure peuvent aller jusqu'à 365 jours à compter de la date de la requête.

```
$ aws kms import-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
    --import-token fileb://ImportToken.bin \
    --expiration-model KEY_MATERIAL_EXPIRES \
    --valid-to 2023-06-17T12:00:00-08:00
```

   Si vous souhaitez que l'élément de clé importé n'expire pas, définissez la valeur du paramètre `expiration-model` sur `KEY_MATERIAL_DOES_NOT_EXPIRE`, et omettez le paramètre `valid-to` de la commande.

```
$ aws kms import-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
    --import-token fileb://ImportToken.bin \
    --expiration-model KEY_MATERIAL_DOES_NOT_EXPIRE
```

5. Si vous souhaitez importer de nouveaux éléments clés, non associés auparavant à la clé KMS, définissez le `ImportType` paramètre sur`NEW_KEY_MATERIAL`. Cette option ne peut être utilisée qu'avec des clés de chiffrement symétriques à région unique. Pour ces clés, vous pouvez également utiliser le `KeyMaterialDescription` paramètre facultatif pour définir une description du contenu clé importé dans l'exemple de ligne de commande suivant :

```
$ aws kms import-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
    --import-token fileb://ImportToken.bin \
    --expiration-model KEY_MATERIAL_EXPIRES \
    --valid-to 2023-06-17T12:00:00-08:00 \
    --import-type NEW_KEY_MATERIAL \
    --key-material-description "Q2 2025 Rotation"
```

> **Tip**
>
> Si la commande échoue, vous pouvez voir un `KMSInvalidStateException` ou un `NotFoundException`. Vous pouvez réessayer la demande.

Création d'une clé KMS dans un magasin de AWS CloudHSM clés

Après avoir créé un magasin de AWS CloudHSM clés, vous pouvez le créer AWS KMS keys dans votre magasin de clés. Il doit s'agir de clés KMS de chiffrement symétriques dont le contenu clé est AWS KMS généré. Vous ne pouvez pas créer de clés KMS asymétriques, de clés KMS HMAC ou de clés KMS avec des éléments de clé importés dans un magasin de clé personnalisé. De plus, vous ne pouvez pas utiliser de clés KMS de chiffrement symétriques dans un magasin de clés personnalisé pour générer des paires de clés de données asymétriques. KMS ne peut pas communiquer IPv6 avec les magasins de AWS CloudHSM clés.

Pour créer une clé KMS dans un magasin de AWS CloudHSM clés, le magasin de AWS CloudHSM clés doit être connecté au AWS CloudHSM cluster associé et le cluster doit en contenir au

moins deux actifs HSMs dans des zones de disponibilité différentes. Pour connaître l'état de la connexion et le nombre de HSMs, consultez la page des magasins de AWS CloudHSM clés dans le AWS Management Console. Lorsque vous utilisez les opérations d'API, utilisez l'DescribeCustomKeyStoresopération pour vérifier que le magasin de AWS CloudHSM clés est connecté. Pour vérifier le nombre de personnes actives HSMs dans le cluster et leurs zones de disponibilité, utilisez l' AWS CloudHSM DescribeClustersopération.

Lorsque vous créez une clé KMS dans votre magasin de AWS CloudHSM clés, AWS KMS crée la clé KMS dans AWS KMS. Mais il crée le matériau clé pour la clé KMS dans le AWS CloudHSM cluster associé. Plus précisément, se AWS KMS connecte au cluster en tant que `kmsuser`CU que vous avez créé. Ensuite, il crée une clé symétrique AES 256 bits persistante, non extractible dans le cluster. AWS KMS définit la valeur de l'attribut de l'étiquette de clé, qui est visible uniquement dans le cluster, sur l'Amazon Resource Name (ARN) de la clé KMS.

Lorsque la commande réussit, l'état de la clé de la nouvelle clé KMS est `Enabled` et son origine est `AWS_CLOUDHSM`. Vous ne pouvez pas modifier l'origine d'une clé KMS après l'avoir créée. Lorsque vous affichez une clé KMS dans un magasin de AWS CloudHSM clés de la AWS KMS console ou en utilisant l'DescribeKeyopération, vous pouvez voir des propriétés typiques, telles que son identifiant de clé, son état de clé et sa date de création. Mais vous pouvez également voir l'ID du magasin de clés personnalisé et l'ID du cluster AWS CloudHSM (facultatif).

Si votre tentative de création d'une clé KMS dans votre banque de AWS CloudHSM clés échoue, utilisez le message d'erreur pour en déterminer la cause. Cela peut indiquer que le magasin de AWS CloudHSM clés n'est pas connecté (`CustomKeyStoreInvalidStateException`) ou HSMs que le AWS CloudHSM cluster associé ne possède pas les deux clés actives requises pour cette opération (`CloudHsmClusterInvalidConfigurationException`). Pour obtenir de l'aide, consultez Dépannage d'un magasin de clés personnalisé.

Pour un exemple du AWS CloudTrail journal de l'opération qui crée une clé KMS dans un magasin de AWS CloudHSM clés, consultezCreateKey.

Créez une nouvelle clé KMS dans votre magasin de clés CloudHSM

Vous pouvez créer une clé KMS de chiffrement symétrique dans votre magasin de AWS CloudHSM clés de la AWS KMS console ou en utilisant l'CreateKeyopération.

Utilisation de la AWS KMS console

Utilisez la procédure suivante pour créer une clé KMS de chiffrement symétrique dans un magasin de AWS CloudHSM clés.

> **Note**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez Create key.
5. Choisissez Symmetric (Symétrique).
6. Dans Key usage (Utilisation de la clé), l'option Encrypt and decrypt (Chiffrer et déchiffrer) est sélectionnée pour vous. Ne la modifiez pas.
7. Choisissez Options avancées.
8. Dans le champ Key material origin (Origine de la clé), sélectionnez AWS CloudHSM key store (Magasin de clés).

   Vous ne pouvez pas créer de clé multirégionale dans un magasin de AWS CloudHSM clés.
9. Choisissez Suivant.
10. Sélectionnez un magasin de AWS CloudHSM clés pour votre nouvelle clé KMS. Pour créer un nouveau magasin de AWS CloudHSM clés, choisissez Créer un magasin de clés personnalisé.

    Le magasin de AWS CloudHSM clés que vous sélectionnez doit avoir le statut Connecté. Son AWS CloudHSM cluster associé doit être actif et en contenir au moins deux actifs HSMs dans des zones de disponibilité différentes.

    Pour obtenir de l'aide sur la connexion d'un magasin de AWS CloudHSM clés, consultezDéconnecter un magasin de AWS CloudHSM clés. Pour obtenir de l'aide concernant l'ajout HSMs, consultez la section Ajout d'un HSM dans le guide de AWS CloudHSM l'utilisateur.
11. Choisissez Suivant.
12. Saisissez un alias et éventuellement une description pour la clé KMS.

13. (Facultatif). Sur la page Ajouter des identifications, ajoutez des identifications qui identifient ou catégorisent votre clé KMS.

    Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.

14. Choisissez Suivant.

15. Dans la section Administrateurs de clé, sélectionnez les utilisateurs et les rôles IAM qui peuvent gérer la clé KMS. Pour plus d'informations, veuillez consulter la rubrique Autorise les administrateurs de clé à administrer la clé KMS.

    > **Remarques**
    >
    > Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles IAM l'autorisation d'utiliser la clé KMS.
    > Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
    > La AWS KMS console ajoute des administrateurs clés à la politique clé sous l'identifiant de l'instruction`"Allow access for Key Administrators"`. La modification de cet identifiant d'instruction peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

16. (Facultatif) Pour empêcher les administrateurs de clé de supprimer cette clé KMS, décochez la case en bas de la page pour Autoriser les administrateurs de clé à supprimer cette clé.

17. Choisissez Suivant.

18. Dans la section Ce compte, sélectionnez les utilisateurs et les rôles IAM autorisés à utiliser la clé KMS dans le cadre d'opérations cryptographiques. Compte AWS Pour plus d'informations, veuillez consulter la rubrique Allows key users to use the KMS key (Autorise les utilisateurs de clé à utiliser la clé KMS).

> **Remarques**
>
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.
> La AWS KMS console ajoute des utilisateurs clés à la politique clé sous les identificateurs de déclaration `"Allow use of the key"` et`"Allow attachment of persistent resources"`. La modification de ces identificateurs de déclaration peut avoir un impact sur la façon dont la console affiche les mises à jour que vous apportez à l'instruction.

19. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour ce faire, dans la Comptes AWS section Autre au bas de la page, choisissez Ajouter un autre compte Compte AWS et entrez l' Compte AWS identifiant d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

> **Note**
>
> Les administrateurs de l'autre Comptes AWS doivent également autoriser l'accès à la clé KMS en créant des politiques IAM pour leurs utilisateurs. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

20. Choisissez Suivant.
21. Passez en revue les principales déclarations de politique pour trouver la clé. Pour apporter des modifications à la politique clé, sélectionnez Modifier.
22. Choisissez Suivant.
23. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
24. Lorsque vous avez terminé, choisissez Finish (Terminer) pour créer la clé.

Lorsque la procédure aboutit, l'écran affiche la nouvelle clé KMS dans le magasin de AWS CloudHSM clés que vous avez choisi. Lorsque vous choisissez le nom ou l'alias de la nouvelle clé

KMS, l'onglet Configuration cryptographique de sa page détaillée affiche l'origine de la clé KMS (AWS CloudHSM), le nom, l'ID et le type du magasin de clés personnalisé, ainsi que l'ID du AWS CloudHSM cluster. Si la procédure échoue, un message d'erreur s'affiche qui décrit l'échec.

> **Tip**
>
> Pour faciliter l'identification des clés KMS dans un magasin de clés personnalisé, sur la page Clés gérées par le client, ajoutez la colonne Custom key store ID (ID du magasin de clés personnalisé) à l'affichage. Cliquez sur l'icône des paramètres en haut à droite et sélectionnez ID du magasin de clés personnalisé. Pour en savoir plus, consultez Personnalisez l'affichage de votre console.

Utilisation de l' AWS KMS API

Pour créer une nouvelle AWS KMS key (clé KMS) dans votre magasin de AWS CloudHSM clés, utilisez l'CreateKeyopération. Utilisez le paramètre `CustomKeyStoreId` pour identifier votre magasin de clés personnalisé et spécifier une valeur `Origin` égale à `AWS_CLOUDHSM`.

Vous pouvez également souhaiter utiliser le paramètre `Policy` pour spécifier une politique de clé. Vous pouvez modifier la politique clé (PutKeyPolicy) et ajouter des éléments facultatifs, tels qu'une description et des balises à tout moment.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

L'exemple suivant commence par un appel à l'DescribeCustomKeyStoresopération visant à vérifier que le magasin de AWS CloudHSM clés est connecté au AWS CloudHSM cluster associé. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Pour décrire uniquement un magasin de AWS CloudHSM clés en particulier, utilisez son `CustomKeyStoreName` paramètre `CustomKeyStoreId` or (mais pas les deux).

Avant d'exécuter cette commande, remplacez l'exemple d'ID de magasin de clés personnalisé par un ID valide.

> **Note**
>
> N'incluez pas d'informations confidentielles ou sensibles dans les champs `Description` ou `Tags`. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
   "CustomKeyStores": [
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleKeyStore",
      "CustomKeyStoreType": "AWS CloudHSM key store",
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "CONNECTED"
   ],
}
```

L'exemple de commande suivant utilise l'DescribeClustersopération pour vérifier que le AWS CloudHSM cluster associé au `ExampleKeyStore` (cluster-1a23b4cdefg) en possède au moins deux actifs. HSMs Si le cluster en compte moins de deux HSMs, l'`CreateKey`opération échoue.

```
$ aws cloudhsmv2 describe-clusters
{
    "Clusters": [
        {
            "SubnetMapping": {
               ...
            },
            "CreateTimestamp": 1507133412.351,
            "ClusterId": "cluster-1a23b4cdefg",
            "SecurityGroup": "sg-865af2fb",
            "HsmType": "hsm1.medium",
            "VpcId": "vpc-1a2b3c4d",
            "BackupPolicy": "DEFAULT",
            "Certificates": {
                "ClusterCertificate": "-----BEGIN CERTIFICATE-----\...\n-----END
 CERTIFICATE-----\n"
            },
```

```
            "Hsms": [
                {
                    "AvailabilityZone": "us-west-2a",
                    "EniIp": "10.0.1.11",
                    "ClusterId": "cluster-1a23b4cdefg",
                    "EniId": "eni-ea8647e1",
                    "StateMessage": "HSM created.",
                    "SubnetId": "subnet-a6b10bd1",
                    "HsmId": "hsm-abcdefghijk",
                    "State": "ACTIVE"
                },
                {
                    "AvailabilityZone": "us-west-2b",
                    "EniIp": "10.0.0.2",
                    "ClusterId": "cluster-1a23b4cdefg",
                    "EniId": "eni-ea8647e1",
                    "StateMessage": "HSM created.",
                    "SubnetId": "subnet-b6b10bd2",
                    "HsmId": "hsm-zyxwvutsrqp",
                    "State": "ACTIVE"
                },
            ],
            "State": "ACTIVE"
        }
    ]
}
```

Cet exemple de commande utilise l'CreateKeyopération pour créer une clé KMS dans un magasin de AWS CloudHSM clés. Pour créer une clé KMS dans un magasin de AWS CloudHSM clés, vous devez fournir l'ID de magasin de clés personnalisé du magasin de AWS CloudHSM clés et spécifier une `Origin` valeur de`AWS_CLOUDHSM`.

La réponse inclut le magasin IDs de clés personnalisé et le AWS CloudHSM cluster.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de magasin de clés personnalisé par un ID valide.

```
$ aws kms create-key --origin AWS_CLOUDHSM --custom-key-store-id cks-1234567890abcdef0
{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
    "CreationDate": 1.499288695918E9,
    "Description": "Example key",
    "Enabled": true,
    "MultiRegion": false,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "Origin": "AWS_CLOUDHSM"
    "CloudHsmClusterId": "cluster-1a23b4cdefg",
    "CustomKeyStoreId": "cks-1234567890abcdef0"
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
        "SYMMETRIC_DEFAULT"
    ]
  }
}
```


Création d'une clé KMS dans des magasins de clés externes

Après avoir créé et connecté votre magasin de clés externe, vous pouvez le créer AWS KMS keys dans votre magasin de clés. Il doit s'agir de clés KMS de chiffrement symétrique dont la valeur d'origine est External key store (Magasin de clés externe) (`EXTERNAL_KEY_STORE`). Vous ne pouvez pas créer de clés KMS asymétriques, de clés KMS HMAC ou de clés KMS avec des éléments de clé importés dans un magasin de clé personnalisé. De plus, vous ne pouvez pas utiliser de clés KMS de chiffrement symétriques dans un magasin de clés personnalisé pour générer des paires de clés de données asymétriques.

Une clé KMS dans un magasin de clés externe peut présenter une latence, une durabilité et une disponibilité inférieures à celles d'une clé KMS standard, car elle dépend de composants situés à l'extérieur d' AWS. Avant de créer ou d'utiliser une clé KMS dans un magasin de clés externe, vérifiez que vous avez besoin d'une clé dotée de propriétés de magasin de clés externe.

> **Note**
>
> Certains gestionnaires de clés externes proposent une méthode plus simple pour créer des clés KMS dans un magasin de clés externe. Pour en savoir plus, veuillez consulter la documentation de votre gestionnaire de clés externe.

Pour créer une clé KMS dans votre magasin de clés externe, vous devez spécifier les éléments suivants :

- L'ID de votre magasin de clés externe.
- Une origine des éléments de clé du magasin de clés externe (`EXTERNAL_KEY_STORE`).
- L'ID d'une clé externe existante dans le gestionnaire de clés externe associé à votre magasin de clés externe. Cette clé externe fait office d'éléments de clé pour la clé KMS. Vous ne pouvez pas modifier l'ID de clé externe une fois que vous avez créé la clé KMS.

  AWS KMS fournit l'ID de clé externe à votre proxy de stockage de clés externe dans les demandes d'opérations de chiffrement et de déchiffrement. AWS KMS ne peut pas accéder directement à votre gestionnaire de clés externe ou à l'une de ses clés cryptographiques.

Outre la clé externe, une clé KMS dans un magasin de clés externe contient également du matériel AWS KMS clé. Toutes les données chiffrées sous la clé KMS sont d'abord cryptées à l' AWS KMS aide du contenu de la AWS KMS clé, puis par votre gestionnaire de clés externe à l'aide de votre clé externe. Ce processus de double chiffrement garantit que le texte chiffré protégé par une clé KMS dans un magasin de clés externe est au moins aussi robuste que le texte chiffré protégé uniquement par AWS KMS. Pour plus de détails, consultez Fonctionnement des magasins de clés externes.

Lorsque l'opération `CreateKey` aboutit, l'état de clé de la nouvelle clé KMS est `Enabled`. Lorsque vous consultez une clé KMS dans un magasin de clés externe, vous pouvez afficher les propriétés classiques, comme son ID de clé, sa spécification de clé, son utilisation de clé, son état de clé et sa date de création. Mais vous pouvez également voir l'ID et l'état de connexion du magasin de clés externe ainsi que l'ID de la clé externe.

Si votre tentative de créer une clé KMS dans votre magasin de clés externe échoue, utilisez le message d'erreur pour identifier la cause. Il peut indiquer que le magasin de clés externe n'est pas connecté (`CustomKeyStoreInvalidStateException`), que le proxy de votre magasin de clés externe ne trouve pas de clé externe avec l'ID de clé externe spécifié (`XksKeyNotFoundException`) ou que la clé externe est déjà associée à une clé KMS dans le même magasin de clés externe `XksKeyAlreadyInUseException`.

Pour un exemple du AWS CloudTrail journal de l'opération qui crée une clé KMS dans un magasin de clés externe, consultezCreateKey.

Rubriques

- Exigences relatives à une clé KMS dans un magasin de clés externe

- Créez une nouvelle clé KMS dans votre magasin de clés externe

Exigences relatives à une clé KMS dans un magasin de clés externe

Pour créer une clé KMS dans un magasin de clés externe, les propriétés suivantes sont requises pour le magasin de clés externe, la clé KMS et la clé externe qui fait office d'éléments de clé cryptographique externe pour la clé KMS.

Exigences relatives au magasin de clés externe

- Doit être connecté à son proxy de magasin de clés externe.

  Pour consulter l'état de connexion de votre magasin de clés externe, veuillez consulter la rubrique Afficher les magasins de clés externes. Pour connecter votre magasin de clés externe, veuillez consulter la rubrique Connecter et déconnecter les magasins de clés externes.

Exigences relatives aux clés KMS

Vous ne pouvez pas modifier ces propriétés après la création de la clé KMS.

- Spécification de clé : SYMMETRIC_DEFAULT
- Utilisation de clé : ENCRYPT_DECRYPT
- Origine des éléments de clé : EXTERNAL_KEY_STORE
- Multi-région : FALSE

Exigences relatives aux clés externes

- Clé cryptographique AES 256 bits (256 bits aléatoires). La propriété `KeySpec` de la clé externe doit être `AES_256`.
- Activé et disponible pour utilisation. La propriété `Status` de la clé externe doit être `ENABLED`.
- Configuré pour le chiffrement et le déchiffrement. La propriété `KeyUsage` de la clé externe doit inclure `ENCRYPT` et `DECRYPT`.
- Utilisé uniquement avec cette clé KMS. Chaque `KMS key` d'un magasin de clés externe doit être associée à une clé externe différente.

  AWS KMS recommande également que la clé externe soit utilisée exclusivement pour le magasin de clés externe. Cette restriction facilite l'identification et la résolution des problèmes liés à la clé.

- Accessible par le proxy de magasin de clés externe pour le magasin de clés externe.

  Si le proxy de magasin de clés externe ne trouve pas la clé à l'aide de l'ID de clé externe spécifié, l'opération `CreateKey` échoue.
- Peut gérer le trafic prévu Services AWS généré par votre utilisation. AWS KMS recommande que les clés externes soient préparées pour traiter jusqu'à 1 800 demandes par seconde.

Créez une nouvelle clé KMS dans votre magasin de clés externe

Vous pouvez créer une nouvelle clé KMS dans votre magasin de clés externe dans la AWS KMS console ou en utilisant l'CreateKeyopération.

Utilisation de la AWS KMS console

Il existe deux manières de créer une clé KMS dans un magasin de clés externe.

- Méthode 1 (recommandée) : choisissez un magasin de clés externe, puis créez une clé KMS dans ce magasin de clés externe.
- Méthode 2 : créez une clé KMS, puis indiquez qu'elle se trouve dans un magasin de clés externe.

Si vous utilisez la méthode 1, dans laquelle vous choisissez votre magasin de clés externe avant de créer votre clé, choisissez AWS KMS toutes les propriétés de clé KMS requises pour vous et renseignez l'ID de votre magasin de clés externe. Cette méthode évite les erreurs que vous pourriez commettre lors de la création de votre clé KMS.

> **Note**
>
> N'incluez pas d'informations confidentielles ou sensibles dans l'alias, la description ou les balises. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

Méthode 1 (recommandée) : démarrer dans votre magasin de clés externe

Pour utiliser cette méthode, choisissez votre magasin de clés externe, puis créez une clé KMS. La AWS KMS console choisit pour vous toutes les propriétés requises et renseigne l'ID de votre banque de clés externe. Cette méthode évite les nombreuses erreurs que vous pourriez commettre lors de la création de votre clé KMS.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Choisissez le nom de votre magasin de clés externe.
5. Dans le coin supérieur droit, choisissez Create a KMS key in this key store (Créer une clé KMS dans ce magasin de clés).

   Si le magasin de clés externe n'est pas connecté, vous serez invité à le connecter. Si la tentative de connexion échoue, vous devez résoudre le problème et connecter le magasin de clés externe avant de pouvoir y créer une clé KMS.

   Si le magasin de clés externe est connecté, vous êtes redirigé vers la page Customer managed keys (Clés gérées par le client) pour créer une clé. Les valeurs de Key configuration (Configuration de clé) requises sont déjà choisies pour vous. En outre, l'ID du magasin de clés personnalisé de votre magasin de clés externe est renseigné, bien que vous puissiez le modifier.
6. Saisissez l'ID de clé d'une clé externe dans votre gestionnaire de clés externe. Cette clé externe doit remplir les conditions requises pour être utilisée avec une clé KMS. Vous ne pouvez pas modifier cette valeur après la création de la clé.

   Si la clé externe en possède plusieurs IDs, entrez l'ID de clé utilisé par le proxy de stockage de clés externe pour identifier la clé externe.
7. Confirmez que vous avez l'intention de créer une clé KMS dans le magasin de clés externe spécifié.
8. Choisissez Suivant.

   Le reste de cette procédure est identique à la création d'une clé KMS standard.
9. Saisissez un alias (obligatoire) et une description (facultative) pour la clé KMS.
10. (Facultatif). Sur la page Ajouter des identifications, ajoutez des identifications qui identifient ou catégorisent votre clé KMS.

    Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être

utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.

11. Choisissez Suivant.
12. Dans la section Administrateurs de clé, sélectionnez les utilisateurs et les rôles IAM qui peuvent gérer la clé KMS. Pour plus d'informations, veuillez consulter la rubrique Autorise les administrateurs de clé à administrer la clé KMS.

> **Note**
>
> Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles IAM l'autorisation d'utiliser la clé KMS.
> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

13. (Facultatif) Pour empêcher les administrateurs de clé de supprimer cette clé KMS, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).

    La suppression d'une clé KMS est une opération destructrice et irréversible, qui peut rendre le texte chiffré irrécupérable. Vous ne pouvez pas recréer une clé KMS symétrique dans un magasin de clés externe, même si vous disposez des éléments de clé externe. Cependant, la suppression d'une clé KMS n'a aucun effet sur la clé externe qui lui est associée. Pour plus d'informations sur la suppression d'une clé KMS d'un magasin de clés externe, consultez la section Considérations spéciales relatives à la suppression de clés.
14. Choisissez Suivant.
15. Dans la section Ce compte, sélectionnez les utilisateurs et les rôles IAM autorisés à utiliser la clé KMS dans le cadre d'opérations cryptographiques. Compte AWS Pour plus d'informations, veuillez consulter la rubrique Allows key users to use the KMS key (Autorise les utilisateurs de clé à utiliser la clé KMS).

> **Note**
>
> Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles IAM l'autorisation d'utiliser la clé KMS.

> Les bonnes pratiques IAM déconseillent d'avoir recours à des utilisateurs IAM dotés d'informations d'identification à long terme. Dans la mesure du possible, utilisez des rôles IAM, qui fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Security best practices in IAM (Bonnes pratiques de sécurité dans IAM) dans le Guide de l'utilisateur IAM.

16. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour ce faire, dans la Comptes AWS section Autre au bas de la page, choisissez Ajouter un autre compte Compte AWS et entrez l' Compte AWS identifiant d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

> **Note**
>
> Les administrateurs de l'autre Comptes AWS doivent également autoriser l'accès à la clé KMS en créant des politiques IAM pour leurs utilisateurs. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

17. Choisissez Suivant.
18. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
19. Lorsque vous avez terminé, choisissez Finish (Terminer) pour créer la clé.

Méthode 2 : démarrer avec les clés gérées par le client

Cette procédure est identique à la procédure de création d'une clé de chiffrement symétrique avec du matériel AWS KMS clé. Mais, dans le cadre de cette procédure, vous spécifiez l'ID du magasin de clés personnalisé du magasin de clés externe et l'ID de la clé externe. Vous devez également spécifier les valeurs de propriété requises pour une clé KMS dans un magasin de clés externe, telles que la spécification de clé et l'utilisation de la clé.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.

4. Choisissez Create key.
5. Choisissez Symmetric (Symétrique).
6. Dans Key usage (Utilisation de la clé), l'option Encrypt and decrypt (Chiffrer et déchiffrer) est sélectionnée pour vous. Ne la modifiez pas.
7. Choisissez Options avancées.
8. Pour Key material origin (Origine des éléments de clé), choisissez External key store (Magasin de clés externe).
9. Confirmez que vous avez l'intention de créer une clé KMS dans le magasin de clés externe spécifié.
10. Choisissez Suivant.
11. Choisissez la ligne qui représente le magasin de clés externe pour votre nouvelle clé KMS.

    Vous ne pouvez pas choisir un magasin de clés externe déconnecté. Pour connecter un magasin de clés déconnecté, choisissez le nom du magasin de clés, puis, dans Key store actions (Actions du magasin de clés), choisissez Connect (Connecter). Pour plus de détails, consultez Utilisation de la AWS KMS console.
12. Saisissez l'ID de clé d'une clé externe dans votre gestionnaire de clés externe. Cette clé externe doit remplir les conditions requises pour être utilisée avec une clé KMS. Vous ne pouvez pas modifier cette valeur après la création de la clé.

    Si la clé externe en possède plusieurs IDs, entrez l'ID de clé utilisé par le proxy de stockage de clés externe pour identifier la clé externe.
13. Choisissez Suivant.

    Le reste de cette procédure est identique à la création d'une clé KMS standard.
14. Saisissez un alias et éventuellement une description pour la clé KMS.
15. (Facultatif). Sur la page Ajouter des identifications, ajoutez des identifications qui identifient ou catégorisent votre clé KMS.

    Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.
16. Choisissez Suivant.

17. Dans la section Administrateurs de clé, sélectionnez les utilisateurs et les rôles IAM qui peuvent gérer la clé KMS. Pour plus d'informations, veuillez consulter la rubrique Autorise les administrateurs de clé à administrer la clé KMS.

    > **Note**
    >
    > Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles IAM l'autorisation d'utiliser la clé KMS.

18. (Facultatif) Pour empêcher les administrateurs de clé de supprimer cette clé KMS, décochez la case Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).

    La suppression d'une clé KMS est une opération destructrice et irréversible, qui peut rendre le texte chiffré irrécupérable. Vous ne pouvez pas recréer une clé KMS symétrique dans un magasin de clés externe, même si vous disposez des éléments de clé externe. Cependant, la suppression d'une clé KMS n'a aucun effet sur la clé externe qui lui est associée. Pour plus d'informations sur la suppression d'une clé KMS d'un magasin de clés externe, veuillez consulter la rubrique Supprimer un AWS KMS key.

19. Choisissez Suivant.

20. Dans la section Ce compte, sélectionnez les utilisateurs et les rôles IAM autorisés à utiliser la clé KMS dans le cadre d'opérations cryptographiques. Compte AWS Pour plus d'informations, veuillez consulter la rubrique Allows key users to use the KMS key (Autorise les utilisateurs de clé à utiliser la clé KMS).

    > **Note**
    >
    > Les politiques IAM peuvent accorder à d'autres utilisateurs et rôles IAM l'autorisation d'utiliser la clé KMS.

21. (Facultatif) Vous pouvez autoriser d'autres Comptes AWS utilisateurs à utiliser cette clé KMS pour des opérations cryptographiques. Pour ce faire, dans la Comptes AWS section Autre au bas de la page, choisissez Ajouter un autre compte Compte AWS et entrez l' Compte AWS identifiant d'un compte externe. Pour ajouter plusieurs comptes externes, répétez cette étape.

> **Note**
>
> Les administrateurs de l'autre Comptes AWS doivent également autoriser l'accès à la clé KMS en créant des politiques IAM pour leurs utilisateurs. Pour de plus amples informations, veuillez consulter Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS.

22. Choisissez Suivant.
23. Passez en revue les paramètres de clé que vous avez choisis. Vous pouvez toujours revenir en arrière et modifier tous les paramètres.
24. Lorsque vous avez terminé, choisissez Finish (Terminer) pour créer la clé.

Lorsque la procédure réussit, l'écran affiche la nouvelle clé KMS dans le magasin de clés externe que vous avez choisi. Lorsque vous choisissez le nom ou l'alias de la nouvelle clé KMS, l'onglet Cryptographic configuration (Configuration cryptographique) de sa page de détails affiche l'origine de la clé KMS (External key store [Magasin de clés externe]), le nom, l'ID et le type du magasin de clés personnalisé, et l'ID, l'utilisation de clé et l'état de la clé externe. Si la procédure échoue, un message d'erreur s'affiche qui décrit l'échec. Pour , veuillez consulter la rubrique Résoudre les problèmes liés aux magasins de clés externes.

> **Tip**
>
> Pour faciliter l'identification des clés KMS dans un magasin de clés personnalisé, sur la page Customer managed keys (Clés gérées par le client), ajoutez les colonnes Origin (Origine) et Custom key store ID (ID de magasin de clés personnalisé) à l'affichage. Pour modifier les champs du tableau, cliquez sur l'icône d'engrenage dans le coin supérieur droit de la page. Pour plus de détails, consultez Personnalisez l'affichage de votre console.

Utilisation de l' AWS KMS API

Pour créer une nouvelle clé KMS dans un magasin de clés externe, utilisez l'CreateKeyopération. Les paramètres suivants sont obligatoires :

- La valeur `Origin` doit être `EXTERNAL_KEY_STORE`.

- Le paramètre `CustomKeyStoreId` identifie votre magasin de clés externe. La valeur `ConnectionState` du magasin de clés externe spécifié doit être `CONNECTED`. Pour trouver les valeurs de `CustomKeyStoreId` et de `ConnectionState`, utilisez l'opération `DescribeCustomKeyStores`.
- Le paramètre `XksKeyId` identifie la clé externe. Cette clé externe doit remplir les conditions requises pour être associée à une clé KMS.

Vous pouvez également utiliser n'importe lequel des paramètres facultatifs de l'opération `CreateKey`, tels que les paramètres `Policy` ou Balises.

> **Note**
>
> N'incluez pas d'informations confidentielles ou sensibles dans les champs `Description` ou `Tags`. Ces champs peuvent apparaître en texte brut dans CloudTrail les journaux et autres sorties.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Cet exemple de commande utilise l'CreateKeyopération pour créer une clé KMS dans un magasin de clés externe. La réponse contient les propriétés des clés KMS, l'ID du magasin de clés externe, ainsi que l'ID, l'utilisation et l'état de la clé externe.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de magasin de clés personnalisé par un ID valide.

```
$ aws kms create-key --origin EXTERNAL_KEY_STORE --custom-key-store-
id cks-1234567890abcdef0 --xks-key-id bb8562717f809024
{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2022-12-02T07:48:55-07:00",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "CustomKeyStoreId": "cks-1234567890abcdef0",
    "Description": "",
    "Enabled": true,
```

```
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "EXTERNAL_KEY_STORE",
    "XksKeyConfiguration": {
      "Id": "bb8562717f809024"
    }
  }
}
```

Identifier et afficher les clés

Vous pouvez utiliser l'API AWS Management Consoleor AWS Key Management Service (AWS KMS) pour afficher les informations AWS KMS keys relatives à chaque compte et région, y compris les clés KMS que vous gérez et les clés KMS gérées par AWS.

Rubriques

Trouvez l'ID et l'ARN de la clé

Pour identifier un AWS KMS key, vous pouvez utiliser l'ID de clé ou le nom de ressource Amazon (ARN clé). Dans les opérations de chiffrement, vous pouvez également utiliser le nom d'alias ou l'ARN d'alias.

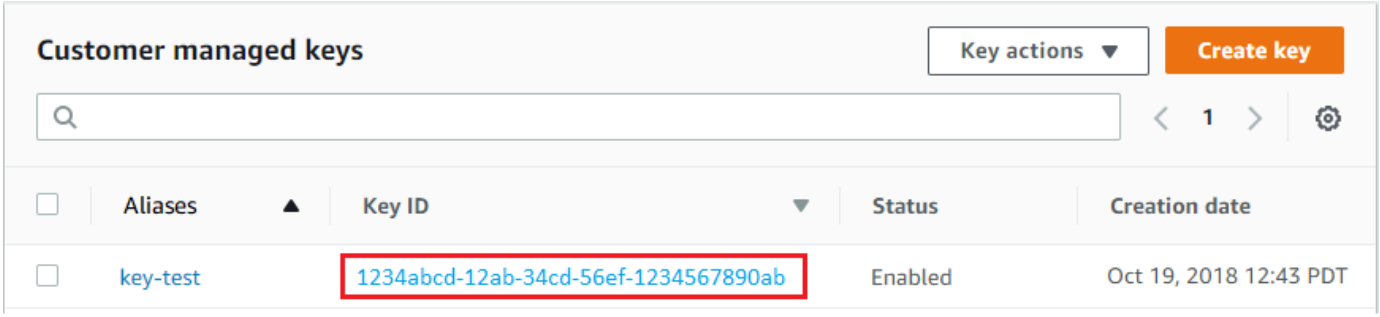
Vous pouvez utiliser la AWS KMS console ou l'ListKeysopération pour identifier l'ID de clé et l'ARN de chaque clé KMS de votre compte et de votre région.

Pour obtenir des informations détaillées sur les identificateurs de clé KMS pris en charge par AWS KMS, consultezIdentifiants clés () KeyId. Pour obtenir de l'aide afin de trouver un nom d'alias et un ARN d'alias, veuillez consulter Trouvez le nom d'alias et l'ARN de l'alias pour une clé KMS.

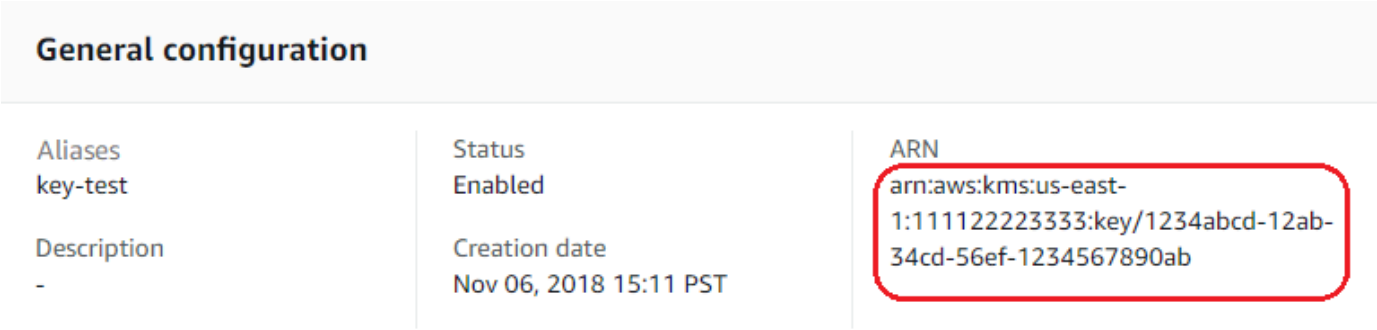
Utilisation de la AWS KMS console

1. Ouvrez la AWS KMS console à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Pour afficher les clés de votre compte que vous créez et gérez vous-même, dans le volet de navigation, choisissez Clés gérées par le client. Pour afficher les clés de votre compte qui AWS crée et gère pour vous, dans le volet de navigation, choisissez les clés AWS gérées.
4. Pour obtenir l'ID de clé d'une clé KMS, veuillez consulter la ligne qui commence par l'alias de clé KMS.

Par défaut, la colonne ID de clé apparaît dans les tables. Si la colonne ID de clé n'apparaît pas dans votre table, utilisez la procédure décrite à la section the section called “Personnalisez l'affichage de votre console” pour la restaurer. Vous pouvez également afficher l'ID de clé d'une clé KMS sur sa page de détails.



5. Pour rechercher l'Amazon Resource Name (ARN) de la clé KMS, choisissez l'ID de clé ou l'alias. L'ARN de clé apparaît dans la section Configuration générale.



Utilisation de l' AWS KMS API

Pour trouver l'ID de clé et l'ARN de clé d'un AWS KMS key, utilisez l'ListKeysopération.

L'`ListKeys`opération renvoie l'ID de clé et le nom de ressource Amazon (ARN) de toutes les clés KMS du compte et de la région de l'appelant.

Par exemple, cet appel à l'opération `ListKeys` renvoie l'ID et l'ARN de chaque clé KMS de ce compte fictif. Pour obtenir des exemples dans plusieurs langages de programmation, veuillez consulter Utilisation `ListKeys` avec un AWS SDK ou une CLI.

```
$ aws kms list-keys
{
    "Keys": [
        {
```

```
            "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
            "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "KeyArn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ]
}
```

Accédez aux informations clés du KMS et listez-les

Vous pouvez utiliser la AWS KMS console ou l'DescribeKeyopération pour accéder aux informations détaillées sur les clés KMS du compte et de la région et les répertorier.

Les procédures suivantes montrent comment accéder aux détails des clés KMS, tels que l'identifiant de la clé, les spécifications de la clé, son utilisation, etc.

Utilisation de la AWS KMS console

La page des détails de chaque clé KMS affiche les propriétés de la clé KMS. Elle diffère légèrement selon les différents types de clés KMS.

Pour afficher des informations détaillées sur une clé KMS, sur la page Clés gérées par AWS ou sur la page des clés gérées par le client, choisissez l'alias ou l'ID de clé de la clé KMS.

La page de détails d'une clé KMS inclut une section General Configuration (Configuration générale) qui affiche les propriétés de base de la clé KMS. Il comprend également des onglets dans lesquels vous pouvez afficher et modifier les propriétés de la clé KMS, telles que la politique de clé, la configuration cryptographique, les balises, le contenu de la clé et les rotations (pour les clés KMS qui prennent en charge la rotation automatique ou à la demande), la régionalité (pour les clés multirégionales) et la clé publique (pour les clés KMS asymétriques).

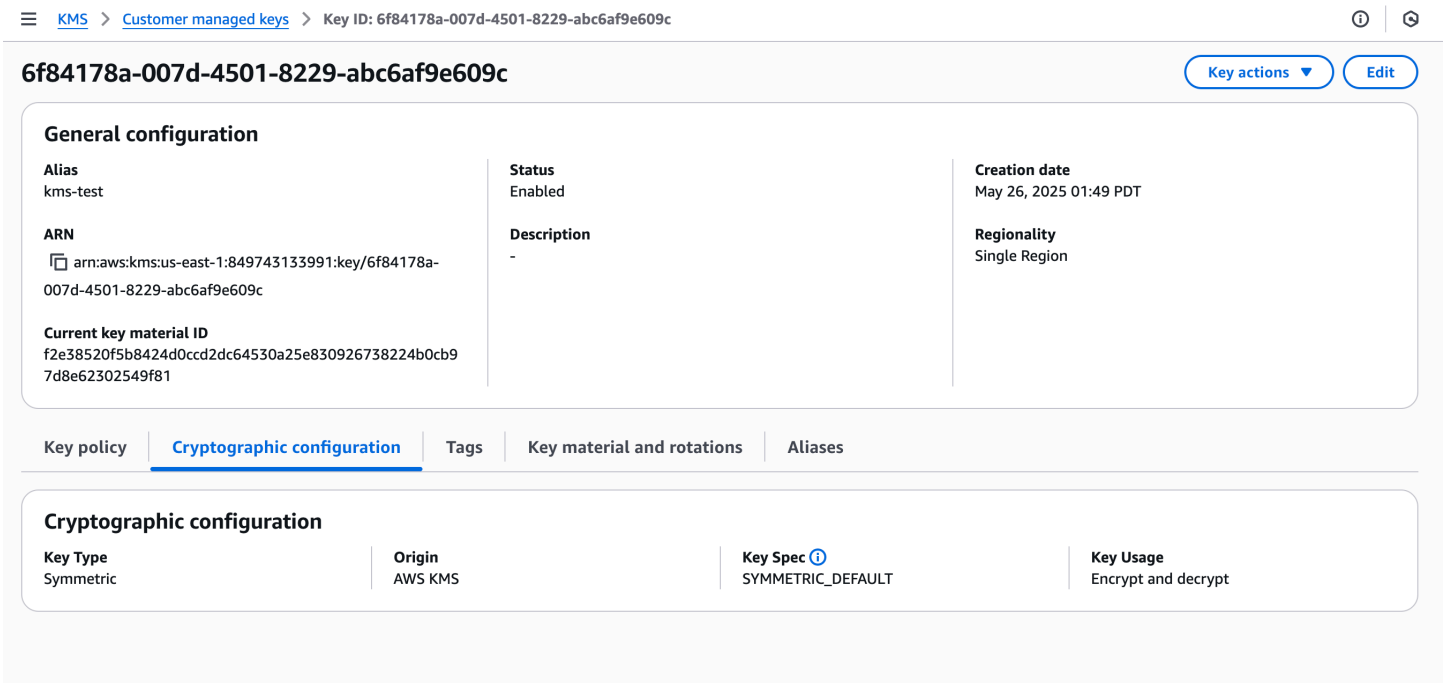
> **Note**
>
> La AWS KMS console affiche les clés KMS que vous êtes autorisé à consulter dans votre compte et dans votre région. Les autres clés KMS Comptes AWS n'apparaissent pas dans la

console, même si vous êtes autorisé à les consulter, à les gérer et à les utiliser. Pour afficher les clés KMS dans d'autres comptes, utilisez l'DescribeKeyopération.

Pour accéder à la page des détails de clé d'une clé KMS.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Pour afficher les clés de votre compte que vous créez et gérez vous-même, dans le volet de navigation, choisissez Clés gérées par le client. Pour afficher les clés de votre compte qui AWS crée et gère pour vous, dans le volet de navigation, choisissez les clés AWS gérées.
4. Pour ouvrir la page des détails de clé, dans la table de clés, choisissez l'ID de clé ou l'alias de la clé KMS.

   Si la clé KMS comporte plusieurs alias, un résumé d'alias (+n plus) apparaît en regard du nom de l'un des alias. Le choix du résumé d'alias vous mène directement à l'onglet Aliases (Alias) dans la page des détails de la clé.



La liste suivante décrit les champs de l'affichage détaillé, y compris les champs dans les onglets. Certains de ces champs sont également disponibles sous forme de colonnes dans l'affichage de la table.

Alias

Où : Onglet Alias

Un nom convivial pour la clé KMS. Vous pouvez utiliser un alias pour identifier la clé KMS dans la console et dans certaines consoles AWS KMS APIs. Pour en savoir plus, consultez Alias dans AWS KMS.

L'onglet Alias affiche tous les alias associés à la clé KMS dans la région Compte AWS et.

ARN

Où : section General configuration (Configuration générale)

Amazon Resource Name (ARN) de la clé KMS. Cette valeur identifie de manière unique la clé KMS. Vous pouvez l'utiliser pour identifier la clé KMS dans les opérations d'API AWS KMS .

État de connexion

Indique si un magasin de clés personnalisé est connecté à son magasin de clés de sauvegarde. Ce champ ne s'affiche que lorsque la clé KMS est créée dans un magasin de clés personnalisé.

Pour plus d'informations sur les valeurs de ce champ, consultez ConnectionStatela référence de l'AWS KMS API.

Date de création

Où : section General configuration (Configuration générale)

Date et heure de création de la clé KMS. Cette valeur est affichée dans l'heure locale du périphérique. Le fuseau horaire ne dépend pas de la région.

Contrairement à Expiration, la création se réfère uniquement à la clé KMS, pas à ses éléments de clé.

ID de cluster CloudHSM

Où : onglet Cryptographic configuration (Configuration de chiffrement)

L'ID du AWS CloudHSM cluster qui contient le matériel clé pour la clé KMS. Ce champ ne s'affiche que lorsque la clé KMS est créée dans un magasin de clés personnalisé.

Si vous choisissez l'ID de cluster CloudHSM, la page Clusters s'ouvre dans la console. AWS CloudHSM

Matériel clé actuel

Où : section General configuration (Configuration générale)

Les clés de chiffrement symétriques avec `AWS_KMS` origine prennent en charge la rotation automatique et à la demande. Les clés de chiffrement symétriques à région unique avec `EXTERNAL` origine prennent en charge la rotation à la demande. Plusieurs matériaux clés peuvent être associés à ces clés. Le matériel clé le plus récemment modifié peut être utilisé à la fois pour le chiffrement et le déchiffrement. Ce matériel clé est identifié comme le matériel clé actuel. Les autres éléments clés ne peuvent être utilisés que pour le déchiffrement. La rotation automatique ou à la demande d'une clé KMS modifie son contenu clé actuel.

ID du magasin de clés personnalisé

Où : onglet Cryptographic configuration (Configuration de chiffrement)

L'ID du magasin de clés personnalisé qui contient la clé KMS. Ce champ ne s'affiche que lorsque la clé KMS est créée dans un magasin de clés personnalisé.

Si vous choisissez l'ID du magasin de clés personnalisé, la page des magasins de clés personnalisés s'ouvre dans la AWS KMS console.

Nom du magasin de clés personnalisé

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Le nom du magasin de clés personnalisé qui contient la clé KMS. Ce champ ne s'affiche que lorsque la clé KMS est créée dans un magasin de clés personnalisé.

Type de magasin de clés personnalisé

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Indique si le magasin de clés personnalisé est un magasin de clés AWS CloudHSM ou un magasin de clés externe. Ce champ ne s'affiche que lorsque la clé KMS est créée dans un magasin de clés personnalisé.

Description

Où : section General configuration (Configuration générale)

Une brève description facultative de la clé KMS que vous pouvez écrire et modifier. Pour ajouter ou mettre à jour la description d'une clé gérée par le client, au-dessus de General Configuration (Configuration générale), choisissez Edit (Modifier).

Algorithmes de chiffrement

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Répertorie les algorithmes de chiffrement qui peuvent être utilisés avec la clé KMS dans AWS KMS. Ce champ s'affiche uniquement lorsque le Type de clé est Asymmetric (Asymétrique) et que Key usage (Utilisation de la clé) est Encrypt and decrypt (Chiffrer et déchiffrer). Pour plus d'informations sur les algorithmes de chiffrement pris AWS KMS en charge, reportez-vous Spécification de clé SYMMETRIC_DEFAULT aux sections etSpécifications des clés RSA pour le chiffrement et le déchiffrement.

Date d’expiration

Où : onglet Key material (Éléments de clé)

La date et l'heure auxquelles les éléments de clé KMS expirent. Ce champ s'affiche uniquement pour les clés KMS avec des éléments de clé importés, c'est-à-dire lorsque l'Origine est Externe et que la clé KMS a des éléments de clé qui expirent. Les clés de chiffrement symétriques à région unique peuvent être associées à plusieurs éléments clés. Pour ces clés, ce champ indique la date et l'heure les plus anciennes auxquelles l'un des éléments clés associés expire.

ID de clé externe

Où : onglet Cryptographic configuration (Configuration de chiffrement)

L'ID de la clé externe associée à une clé KMS dans un magasin de clés externe. Ce champ ne s'affiche que pour les clés KMS dans un magasin de clés externe.

État de la clé externe

Où : onglet Cryptographic configuration (Configuration de chiffrement)

État le plus récent signalé par le proxy de magasin de clés externe pour la clé externe associée à la clé KMS. Ce champ ne s'affiche que pour les clés KMS dans un magasin de clés externe.

Utilisation d'une clé externe

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Opérations cryptographiques activées sur la clé externe associée à la clé KMS. Ce champ ne s'affiche que pour les clés KMS dans un magasin de clés externe.

Stratégie de clé

Où : Onglet Key policy (Politique de clé)

Contrôle l'accès à la clé KMS ainsi qu'aux politiques IAM et aux octrois. Chaque clé KMS a une politique de clé. C'est le seul élément d'autorisation obligatoire. Pour modifier la politique d'une clé KMS gérée par un client, sous l'onglet Key policy (Politique de clé), choisissez Edit (Modifier). Pour en savoir plus, consultez the section called “Politiques de clé”.

Matériau clé et rotations

Où : onglet Matériau clé et rotations

Cet onglet apparaît uniquement pour les clés de chiffrement symétriques avec `AWS_KMS` origine (qui prennent en charge la rotation automatique et à la demande) ainsi que pour les clés de chiffrement symétriques à région unique avec `EXTERNAL` origine (qui prennent en charge la rotation à la demande).

L'onglet comporte trois panneaux :

Rotation automatique : active et désactive la rotation automatique du contenu clé d'une clé KMS gérée par le client. Pour modifier le statut de rotation des clés d'une clé gérée par le client, cochez la case. Vous ne pouvez pas activer ou désactiver la rotation des éléments de clé dans une Clé gérée par AWS. Les Clés gérées par AWS sont automatiquement soumises à la rotation chaque année.

Rotation à la demande : initiez une rotation à la demande du contenu clé d'une clé gérée par le client. Pour les clés importées, un matériau clé importé doit déjà être en `PENDING_ROTATION` état pour que l'option Rotate now soit disponible.

Matériaux clés : répertorie tous les éléments clés associés à la clé KMS. Chaque matériau clé possède un identifiant unique et sa rangée affiche des informations supplémentaires sur le matériau clé, telles que la date de rotation à laquelle le matériau clé est devenu disponible pour être utilisé dans KMS. Pour les clés importées, chaque ligne comporte également un menu Actions qui peut être utilisé pour supprimer un élément clé spécifique ou le réimporter dans la clé KMS.

Spécifications de la clé

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Type de contenu clé contenu dans la clé KMS. AWS KMS prend en charge les clés KMS de chiffrement symétriques (SYMMETRIC_DEFAULT), les clés HMAC KMS de différentes longueurs,

les clés KMS pour les clés RSA de différentes longueurs et les clés à courbe elliptique avec des courbes différentes. Pour en savoir plus, consultez Key spec.

Type de clé

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Indique si la clé KMS est Symmetric (Symétrique) ou Asymmetric (Asymétrique).

Utilisation de la clé

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Indique si une clé KMS peut être utilisée pour Encrypt and decrypt (Chiffrer et déchiffrer), Sign and verify (Signer et vérifier) ou Generate and verify MAC (Générer et vérifier le MAC). Pour en savoir plus, consultez Key usage.

Origin

Où : onglet Cryptographic configuration (Configuration de chiffrement)

La source de l'élément de clé pour la clé KMS. Les valeurs valides sont :

- AWS KMS pour les éléments de clé que AWS KMS génère
- AWS CloudHSM pour les clés KMS dans le magasin de clés AWS CloudHSM
- External (Externe) pour les éléments de clé importés (BYOK)
- External key store (Magasin de clés externe) pour les clés KMS dans un magasin de clés externe

Algorithmes MAC

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Répertorie les algorithmes MAC qui peuvent être utilisés avec la clé KMS HMAC dans AWS KMS. Ce champ apparaît uniquement lorsque la spécification de clé est une spécification de clé HMAC (HMAC_*). Pour de plus amples informations sur les algorithmes MAC pris en charge par AWS KMS , veuillez consulter Spécifications de clé pour les clés KMS HMAC.

Clé primaire

Où : Onglet Regionality (Régionalité)

Indique que cette clé KMS est une clé primaire multi-région. Les utilisateurs autorisés peuvent utiliser cette section pour modifier la clé primaire en une autre clé multi-région associée. Ce champ apparaît uniquement lorsque la clé KMS est une clé principale multi-région.

Clé publique

Où : Onglet Public key (Clé publique)

Affiche la clé publique d'une clé KMS asymétrique. Les utilisateurs autorisés peuvent utiliser cet onglet pour copier et télécharger la clé publique.

Régionalité

Où : section General configuration (Configuration générale) et onglet Regionality (Régionalité)

Indique si une clé KMS est une clé de région unique, une clé primaire multi-région ou une clé de réplica multi-région. Ce champ apparaît uniquement lorsque la clé KMS est une clé multi-région.

Touches multi-région associées

Où : Onglet Regionality (Régionalité)

Affiche tous les clés primaires et de réplica multi-région, à l'exception de la clé KMS actuelle. Ce champ apparaît uniquement lorsque la clé KMS est une clé multi-région.

Dans la section Related multi-Region keys (Clés multi-région associées) d'une clé primaire, les utilisateurs autorisés peuvent créer des clés de réplica.

Clé de réplica

Où : Onglet Regionality (Régionalité)

Indique que cette clé KMS est une clé de réplica multi-région. Ce champ apparaît uniquement lorsque la clé KMS est une clé de réplica multi-région.

Algorithmes de signature

Où : onglet Cryptographic configuration (Configuration de chiffrement)

Répertorie les algorithmes de signature qui peuvent être utilisés avec la clé KMS dans AWS KMS. Ce champ s'affiche uniquement lorsque le Type de clé est Asymmetric (Asymétrique) et que Key usage (Utilisation de la clé) est Sign and verify (Signer et vérifier). Pour plus d'informations sur les algorithmes de signature compatibles AWS KMS , reportez-vous Spécifications des clés RSA pour la signature et la vérification aux sections etSpécifications de la clé de courbe elliptique.

Statut

Où : section General configuration (Configuration générale)

État de clé de la clé KMS. Vous pouvez utiliser la clé KMS dans les opérations de chiffrement uniquement lorsque l'état est Enabled (Activé). Pour obtenir une description détaillée de chaque état de clé KMS et de son effet sur les opérations que vous pouvez exécuter sur la clé KMS, veuillez consulter États clés des AWS KMS clés.

Balises

Où : Onglet Tags (Balises)

Paires clé-valeur facultatives décrivant la clé KMS. Pour ajouter ou modifier les balises d'une clé KMS, sous l'onglet Tags (Balises), choisissez Edit (Modifier).

Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Les balises peuvent également être utilisées pour contrôler l'accès à une clé KMS. Pour de plus amples informations sur l'étiquetage des clés KMS, veuillez consulter Tags dans AWS KMS et ABAC pour AWS KMS.

Utilisation de l' AWS KMS API

L'DescribeKeyopération renvoie des informations sur la clé KMS spécifiée. Pour identifier la clé KMS, utilisez son ID de clé, son ARN de clé, son nom d'alias ou son ARN d'alias.

Contrairement à l'ListKeysopération, qui affiche uniquement les clés KMS dans le compte et la région de l'appelant, les utilisateurs autorisés peuvent utiliser l'`DescribeKey`opération pour obtenir des informations sur les clés KMS d'autres comptes.

> **Note**
>
> La réponse `DescribeKey` inclut à la fois les membres `KeySpec` et `CustomerMasterKeySpec` avec les mêmes valeurs. Le membre `CustomerMasterKeySpec` est obsolète.

Par exemple, cet appel à `DescribeKey` renvoie des informations sur une clé KMS de chiffrement symétrique. Les champs de la réponse varient en fonction des spécifications AWS KMS key, de l'état de la clé et de l'origine des éléments de clé. Pour obtenir des exemples dans plusieurs langages de programmation, veuillez consulter Utilisation `DescribeKey` avec un AWS SDK ou une CLI.

```
$ aws kms describe-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

```
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "Enabled": true,
        "KeySpec": "SYMMETRIC_DEFAULT",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "CreationDate": 1499988169.234,
        "MultiRegion": false,
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "CurrentKeyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    }
}
```

Cet exemple appelle une opération `DescribeKey` sur une clé KMS asymétrique utilisée pour la signature et la vérification. La réponse inclut les algorithmes de signature que AWS KMS prend en charge pour cette clé KMS.

```
$ aws kms describe-key --key-id 0987dcba-09fe-87dc-65ba-ab0987654321

{
    "KeyMetadata": {
        "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "Origin": "AWS_KMS",
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "KeyState": "Enabled",
        "KeyUsage": "SIGN_VERIFY",
        "CreationDate": 1569973196.214,
        "Description": "",
        "KeySpec": "ECC_NIST_P521",
        "CustomerMasterKeySpec": "ECC_NIST_P521",
        "AWSAccountId": "111122223333",
```

```
        "Enabled": true,
        "MultiRegion": false,
        "KeyManager": "CUSTOMER",
        "SigningAlgorithms": [
            "ECDSA_SHA_512"
        ]
    }
}
```

Identifier les différents types de clés

Les rubriques suivantes expliquent comment identifier les différents types de clés dans la AWS KMS console et DescribeKeyles réponses.

Pour obtenir de l'aide pour accéder à l'onglet Configuration cryptographique de la page de détails d'une clé KMS, consultez. the section called “Accédez aux informations clés du KMS et listez-les”

Rubriques

- Identifier les clés KMS asymétriques
- Identifier les clés HMAC KMS
- Identifier les clés KMS multirégionales
- Identifiez les clés KMS avec du matériel clé importé
- Identifiez les clés KMS dans les magasins de AWS CloudHSM clés
- Identifier les clés KMS dans les magasins de clés externes

Identifier les clés KMS asymétriques

Dans la AWS KMS console

La colonne Type de clé du tableau des clés gérées par le client indique si chaque clé KMS est symétrique ou asymétrique. Vous pouvez filtrer le tableau en fonction de la valeur du type de clé pour afficher uniquement les clés KMS asymétriques. Pour de plus amples informations, veuillez consulter the section called “Triez et filtrez vos clés KMS”.

L'onglet Configuration cryptographique de la page de détails d'une clé KMS affiche le type de clé, qui indique si la clé est symétrique ou asymétrique. Il affiche également l'utilisation de la clé, qui indique si votre clé KMS asymétrique est utilisée pour le chiffrement et le déchiffrement, la signature et la vérification, ou pour la dérivation de secrets partagés.

Dans les DescribeKey réponses

Lorsque vous appelez l'`DescribeKey`opération sur une clé KMS asymétrique, la réponse inclut les `KeyUsage` valeurs `KeySpec` et, qui peuvent être utilisées pour déterminer si une clé KMS est symétrique ou asymétrique.

Si la `KeySpec` valeur est`SYMMETRIC_DEFAULT`, la clé est une clé KMS de chiffrement symétrique. Pour plus de détails sur les caractéristiques des clés asymétriques, voirRéférence de spécification clé.

Si la `KeyUsage` valeur est `SIGN_VERIFY` ou`KEY_AGREEMENT`, la clé est une clé KMS asymétrique.

L'`DescribeKey`opération renvoie également les informations suivantes pour les clés KMS asymétriques.

- Pour les clés KMS asymétriques dont `KeyUsage` la valeur est égale à`ENCRYPT_DECRYPT`, l'opération renvoie le`EncryptionAlgorithms`, qui répertorie les algorithmes de chiffrement valides pour la clé.
- Pour les clés KMS asymétriques dont `KeyUsage` la valeur est égale à`SIGN_VERIFY`, l'opération renvoie le`SigningAlgorithms`, qui répertorie les algorithmes de signature valides pour la clé.
- Pour les clés KMS asymétriques dont `KeyUsage` la valeur est égale à`KEY_AGREEMENT`, l'opération renvoie le`KeyAgreementAlgorithms`, qui répertorie les algorithmes d'accord de clé valides pour la clé.

Pour plus d'informations sur les clés KMS asymétriques, consultezthe section called “Clés asymétriques”.

Identifier les clés HMAC KMS

Dans la AWS KMS console

Les clés HMAC KMS sont incluses dans le tableau des clés gérées par le client, mais vous ne pouvez pas trier ou filtrer ce tableau en fonction de la spécification des clés ou des valeurs d'utilisation des clés qui identifient les clés HMAC. Pour faciliter la recherche de vos clés HMAC, attribuez-leur un alias distinctif ou une balise distinctive. Vous pouvez ensuite trier ou filtrer par alias ou balise.

L'onglet Configuration cryptographique de la page de détails d'une clé KMS affiche le type de clé, qui indique si la clé est symétrique ou asymétrique. Les clés KMS HMAC sont symétriques. L'onglet Configuration cryptographique affiche également l'utilisation de la clé. Pour les clés HMAC KMS, la valeur d'utilisation de la clé est toujours Générer et vérifier le MAC.

Dans les DescribeKey réponses

Lorsque vous appelez l'`DescribeKey`opération sur une clé HMAC KMS, la réponse inclut les `KeyUsage` valeurs `KeySpec` et. Pour les clés HMAC KMS, la valeur d'utilisation de la clé est toujours `GENERATE_VERIFY_MAC` et la valeur de spécification de la clé commence toujours par. `HMAC_`

Pour plus d'informations sur les clés HMAC KMS, consultezthe section called “Clés HMAC”.

Identifier les clés KMS multirégionales

Dans la AWS KMS console

Le tableau des clés gérées par le client affiche uniquement les clés KMS dans la région sélectionnée. Vous pouvez afficher les clés principales et de réplica multi-région dans la région sélectionnée. Pour modifier la AWS région, utilisez le sélecteur de région situé dans le coin supérieur droit de la console.

Pour faciliter l'identification des clés multirégionales dans le tableau des clés gérées par le client, ajoutez la colonne Régionalité à votre tableau. Pour obtenir de l'aide, veuillez consulter the section called “Personnalisez vos tableaux de clés KMS”.

La page détaillée des clés KMS multirégionales inclut un onglet Régionalité. L'onglet Regionality (Régionalité) d'une clé principale inclut les boutons Change primary Region (Modifier la région principale) et Create new replica keys (Créer de nouvelles clés de réplica). (L'onglet Regionality (Régionalité) d'une clé de réplica n'a aucun bouton.) La section Related multi-Region keys (Clés multi-région associées) répertorie toutes les clés multi-région associées à la clé actuelle. Si la clé actuelle est une clé de réplica, cette liste inclut la clé principale.

Si vous choisissez une clé multirégionale associée dans le tableau des clés multirégionales associées, la AWS KMS console passe à la région de la clé sélectionnée et ouvre la page détaillée de la clé. Par exemple, si vous choisissez la clé de réplique dans la `sa-east-1` région dans la section d'exemple de clés multirégionales associées ci-dessous, la AWS KMS console passe à la `sa-east-1` région pour afficher la page détaillée de cette clé de réplique. Vous

pouvez le faire pour afficher l'alias ou la politique de clé de la clé de réplica. Pour changer de région à nouveau, utilisez le Sélecteur de région dans l'angle supérieur droit de la page.

Dans les DescribeKey réponses

Par défaut, les opérations d' AWS KMS API sont régionales et ne renvoient que les ressources de la région actuelle ou spécifiée. Toutefois, lorsque vous appelez l'`DescribeKey`opération sur une clé KMS multirégionale, la réponse inclut toutes les clés multirégionales associées dans les autres AWS régions de l'`MultiRegionConfiguration`élément.

Pour plus d'informations sur les clés KMS multirégionales, consultezthe section called “Clés multi-région”.

Identifiez les clés KMS avec du matériel clé importé

Dans la AWS KMS console

Pour faciliter l'identification des clés KMS contenant des éléments clés importés dans le tableau des clés gérées par le client, ajoutez la colonne Origine à votre tableau. La colonne Origine permet d'identifier facilement les clés KMS avec une valeur de propriété d'origine Externe (Importation d’éléments de clé). Pour obtenir de l'aide, veuillez consulter the section called “Personnalisez vos tableaux de clés KMS”.

L'onglet Configuration cryptographique de la page de détails d'une clé KMS affiche l'origine, qui identifie la source du contenu clé de la clé KMS. Pour les clés KMS dont le matériel clé est importé, la valeur d'origine est toujours Externe (matériau clé d'importation). La page de détails inclut également un onglet Matériau clé qui fournit des informations détaillées sur le matériau clé importé. Les clés de chiffrement symétriques à région unique avec `EXTERNAL` origine prennent en charge les rotations à la demande et peuvent être associées à plusieurs éléments clés. Pour ces touches, l'onglet est intitulé Matériau clé et rotations.

Dans les DescribeKey réponses

Lorsque vous appelez l'`DescribeKey`opération sur une clé KMS avec du matériel clé importé`Origin`, la réponse inclut les `ValidTo` valeurs`ExpirationModel`, et. Pour les clés KMS dont le matériel clé est importé, la valeur d'origine est toujours`EXTERNAL`. La `ExpirationModel` valeur indique si le matériau clé doit expirer ou non, et la `ValidTo` valeur indique quand le matériau clé expirera. Lorsque plusieurs documents clés sont associés à une clé, la `ValidTo` valeur indique le délai d'expiration le plus proche pour tous les documents clés (à l'exception

de celui en attente de rotation) et `ExpirationModel` est définie sur `DOES_NOT_EXPIRE` uniquement si aucun de ces matériaux clés n'est configuré pour expirer. Pour de plus amples informations, veuillez consulter Définir un délai d'expiration (facultatif).

Pour plus d'informations sur les clés KMS contenant des éléments clés importés, consultezthe section called “Éléments de clé importés”.

Identifiez les clés KMS dans les magasins de AWS CloudHSM clés

Dans la AWS KMS console

Pour faciliter l'identification des clés KMS dans les magasins de AWS CloudHSM clés du tableau des clés gérées par le client, ajoutez la colonne Origine à votre tableau. La colonne Origin (Origine) permet d'identifier facilement les clés KMS avec une valeur de propriété d'origine AWS CloudHSM. Pour obtenir de l'aide, veuillez consulter the section called “Personnalisez vos tableaux de clés KMS”.

L'onglet Configuration cryptographique de la page de détails d'une clé KMS affiche l'origine, qui identifie la source du contenu clé de la clé KMS. Pour les clés KMS dans les magasins de AWS CloudHSM clés, la valeur d'origine est toujours AWS CloudHSM.

Pour une clé KMS dans un magasin de AWS CloudHSM clés, l'onglet Configuration cryptographique inclut une section supplémentaire, Stockage de clés personnalisé, qui fournit des informations sur le magasin de AWS CloudHSM clés et le AWS CloudHSM cluster associés à la clé KMS.

Dans les DescribeKey réponses

Lorsque vous appelez l'`DescribeKey`opération sur une clé KMS dans un magasin de AWS CloudHSM clés`Origin`, la réponse inclut le, qui identifie la source du matériel clé. Pour les clés KMS d'un magasin de AWS CloudHSM clés, la valeur d'origine est toujours`AWS_CLOUDHSM`. L'opération renvoie également les champs spéciaux suivants pour les clés KMS dans les magasins de AWS CloudHSM clés :

- `CloudHsmClusterId`
- `CustomKeyStoreId`

Pour plus d'informations sur les AWS CloudHSM principaux magasins, consultezthe section called “AWS CloudHSM magasins clés”.

Identifier les clés KMS dans les magasins de clés externes

Dans la AWS KMS console

Pour faciliter l'identification des clés KMS dans les magasins de clés externes dans le tableau des clés gérées par le client, ajoutez la colonne Origine à votre tableau. La colonne Origine permet d'identifier facilement les clés KMS avec une valeur de propriété d'origine du magasin de clés externes. Pour obtenir de l'aide, veuillez consulter the section called “Personnalisez vos tableaux de clés KMS”.

L'onglet Configuration cryptographique de la page de détails d'une clé KMS affiche l'origine, qui identifie la source du contenu clé de la clé KMS. Pour les clés KMS dans les magasins de clés externes, la valeur d'origine est toujours le magasin de clés externe.

Pour une clé KMS dans un magasin de clés externe, l'onglet Configuration cryptographique comprend deux sections supplémentaires, le magasin de clés personnalisé et la clé externe. Le tableau du magasin de clés personnalisé fournit des informations sur le magasin de clés externe associé à la clé KMS. Le tableau des clés externes apparaît dans la AWS KMS console uniquement pour les clés KMS dans les magasins de clés externes. Elle fournit des informations sur la clé externe associée à la clé KMS. La clé externe est une clé cryptographique extérieure AWS qui sert de matériau clé pour la clé KMS dans le magasin de clés externe. Lorsque vous chiffrez ou déchiffrez à l'aide de la clé KMS, l'opération est exécutée par votre gestionnaire de clés externe à l'aide de la clé externe spécifiée.

Les valeurs suivantes apparaissent dans la section External key (Clé externe).

ID de clé externe

L'identifiant de la clé externe dans son gestionnaire de clés externe. Il s'agit de la valeur que le proxy de magasin de clés externe utilise pour identifier la clé externe. Vous choisissez l'ID de la clé externe lorsque vous créez la clé KMS et vous ne pouvez pas le modifier. Si la valeur d'ID de clé externe que vous avez utilisée pour créer la clé KMS change ou devient invalide, vous devez planifier la suppression de la clé KMS et créer une clé KMS avec la valeur d'ID de clé externe correcte.

Dans les DescribeKey réponses

Lorsque vous appelez l'`DescribeKey`opération sur une clé KMS dans un magasin de clés externe`Origin`, la réponse inclut le, qui identifie la source du contenu clé. Pour les clés KMS d'un magasin de AWS CloudHSM clés, la valeur d'origine est toujours`EXTERNAL_KEY_STORE`.

L'opération renvoie également l'`CustomKeyStoreId`élément, qui identifie le magasin de clés externe associé aux clés KMS.

Pour plus d'informations sur les magasins de clés externes, consultezthe section called “Magasins de clés externes”.

Personnalisez l'affichage de votre console

Vous pouvez personnaliser l'affichage de la AWS KMS console pour retrouver plus facilement vos clés KMS. Personnalisez les tableaux qui apparaissent sur les pages Clés gérées par AWSet sur les pages des clés gérées par le client pour afficher les informations dont vous avez le plus besoin, ou triez et filtrez les clés KMS renvoyées dans les tableaux.

Rubriques

- Triez et filtrez vos clés KMS
- Personnalisez vos tableaux de clés KMS

Triez et filtrez vos clés KMS

Pour faciliter la recherche de vos clés KMS dans la console, vous pouvez trier et filtrer les tables de clés.

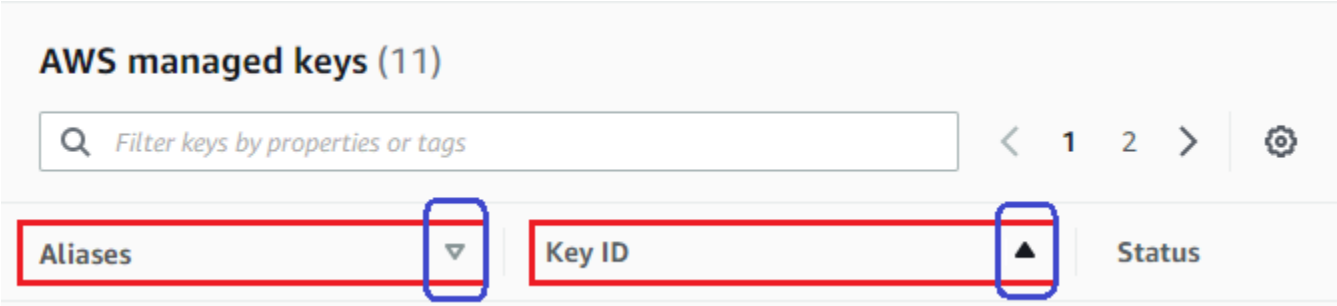
Tri

Vous pouvez trier les clés KMS par ordre croissant ou décroissant selon les valeurs de leurs colonnes. Cette fonction trie toutes les clés KMS de la table, même si elles n'apparaissent pas sur la page de la table en cours.

Les colonnes pouvant être triées sont indiquées par une flèche à côté du nom de la colonne. Dans la page Clés gérées par AWS, vous pouvez trier par Alias ou ID de clé. Sur la page Customer managed keys (Clés gérées par le client), vous pouvez trier par Alias, ID de clé ou Type de clé.

Pour trier par ordre croissant, choisissez l'en-tête de colonne jusqu'à ce que la flèche pointe vers le haut. Pour trier par ordre décroissant, choisissez l'en-tête de colonne jusqu'à ce que la flèche pointe vers le bas. Vous pouvez trier uniquement selon une colonne à la fois.

Par exemple, vous pouvez trier les clés KMS par ordre croissant par ID de clé, au lieu d'alias, qui est la valeur par défaut.



Lorsque vous triez vos clés KMS sur la page Clés gérées par le client dans l'ordre croissant par Type de clé, toutes les clés asymétriques sont affichées avant toutes les clés symétriques.

Filtre

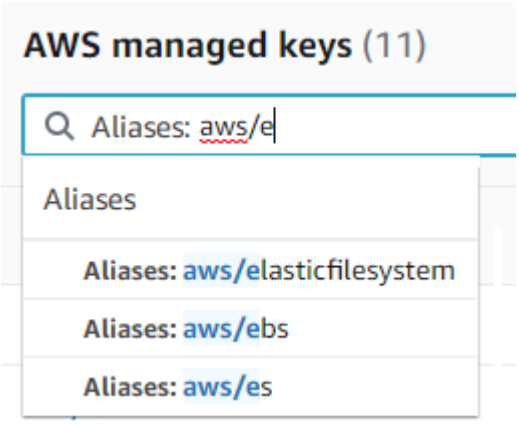
Vous pouvez filtrer les clés KMS en fonction de leurs valeurs de propriété ou de leurs balises. Le filtre s'applique à toutes les clés KMS de la table, même si elles n'apparaissent pas sur la page actuelle de la table. Le filtre n'est pas sensible à la casse.

Les propriétés pouvant être filtrées sont répertoriées dans la zone de filtre. Dans la page Clés gérées par AWS , vous pouvez filtrer par alias et ID de clé. Sur la page Clés gérées par le client, vous pouvez filtrer en fonction de leurs propriétés Alias, ID de clé et Type de clé et en fonction de leurs balises.

- Dans la page Clés gérées par AWS, vous pouvez filtrer par alias et ID de clé.
- Sur la page Clés gérées par le client, vous pouvez filtrer en fonction des balises ou de leurs propriétés Alias, ID de clé et Type de clé et de régionalité.

Pour filtrer en fonction d'une valeur de propriété, choisissez le filtre, le nom de la propriété, puis une valeur dans la liste des valeurs de propriété réelles. Pour filtrer par balise, choisissez la clé de balise, puis choisissez dans la liste des valeurs réelles de balise. Après avoir choisi une clé de propriété ou une clé de balise, vous pouvez également saisir l'ensemble ou une partie de la valeur de propriété ou de la balise. Vous verrez un aperçu des résultats avant de faire votre choix.

Par exemple, pour afficher les clés KMS avec un nom d'alias qui contient `aws/e`, choisissez la zone de filtre, choisissez Alias, saisissez `aws/e`, puis appuyez sur `Enter` ou `Return` pour ajouter le filtre.



Filtres du tableau des clés KMS suggérés

Filtre pour clés KMS asymétriques

Pour afficher uniquement les clés KMS asymétriques sur la page Clés gérées par le client, cliquez sur la zone de filtre, choisissez Key type (Type de clé) puis Key type: Asymmetric (Type de clé : Asymétrique). L'option Asymmetric (Asymétrique) apparaît uniquement lorsque vous avez des clés KMS asymétriques dans le tableau.

Filtre pour clés multirégionales

Pour afficher uniquement les touches multi-région, dans la page Clés gérées par le client, choisissez la zone de filtre, puis Regionality (Régionalité) et Regionality: Multi-Region (Régionalité : Multi-régions). L'option Multi-Region (Multi-régions) apparaît uniquement lorsque vous avez des clés multi-région dans la table.

Filtre pour les tags

Pour afficher uniquement les clés KMS avec une balise particulière, choisissez la zone de filtre, choisissez la clé de balise, puis choisissez parmi les valeurs réelles de balise. Vous pouvez également saisir l'ensemble ou une partie de la valeur de balise.

Le tableau résultant affiche toutes les clés KMS avec la balise choisie. Cependant, il n'affiche pas la balise. Pour afficher la balise, choisissez l'ID de clé ou l'alias de la clé KMS et, sur sa page de détails, choisissez l'option Tags (Balises). Les onglets apparaissent sous la section General configuration (Configuration générale).

Ce filtre nécessite à la fois la clé de balise et la valeur de balise. Il ne trouvera pas de clés KMS en tapant uniquement la clé de balise ou seulement sa valeur. Pour filtrer les

balises en fonction de la totalité ou d'une partie de la clé ou de la valeur de balise, utilisez l'ListResourceTagsopération pour obtenir les clés KMS balisées, puis utilisez les fonctionnalités de filtrage de votre langage de programmation.

Filtrer par texte

Pour rechercher du texte, dans la zone de filtre, saisissez l'ensemble ou une partie d'alias, d'ID de clé, d'un type de clé ou d'une clé de balise. (Après avoir sélectionné la clé de balise, vous pouvez rechercher une valeur de balise). Vous verrez un aperçu des résultats avant de faire votre choix.

Par exemple, pour afficher les clés KMS avec `test` dans ses clés de balise ou ses propriétés filtrables, saisissez `test` dans la zone de filtre. La prévisualisation affiche les clés KMS que le filtre sélectionnera. Dans ce cas, `test` apparaît uniquement dans la propriété Alias.

Personnalisez vos tableaux de clés KMS

Vous pouvez personnaliser les tableaux qui apparaissent sur les pages Clés gérées par AWSdes clés gérées par le client en fonction AWS Management Console de vos besoins. Vous pouvez choisir les colonnes du tableau, le nombre de colonnes AWS KMS keys sur chaque page (taille de page) et l'habillage du texte. La configuration que vous choisissez est enregistrée lorsque vous la confirmez et réappliquée chaque fois que vous ouvrez les pages.

Personnalisation de vos tables de clés KMS

1. Sur la page des Clés gérées par AWS ou des clés gérées par le client, choisissez l'icône des paramètres
   ()
   dans le coin supérieur droit de la page.
2. Sur la page Préférences, choisissez vos paramètres préférés, puis choisissez Confirmer.

Pensez à utiliser le paramètre Page size (Taille de page) pour augmenter le nombre de clés KMS affichées sur chaque page, surtout si vous utilisez généralement un périphérique facile à faire défiler.

Les colonnes de données que vous affichez peuvent varier en fonction de la table, de votre rôle de tâche et des types de clés KMS dans le compte et la région. La table suivante propose quelques configurations. Pour obtenir une description des colonnes, veuillez consulter Utilisation de la AWS KMS console.

Configurations de tables de clés KMS suggérées

Vous pouvez personnaliser les colonnes qui apparaissent dans votre table de clés KMS pour afficher les informations dont vous avez besoin sur vos clés KMS.

Clés gérées par AWS

Par défaut, la table des Clé gérée par AWS affiche les colonnes Alias, Key ID (ID de clé) et Status (État). Ces colonnes sont idéales pour la plupart des cas d'utilisation.

Clés KMS de chiffrement symétriques

Si vous utilisez uniquement des clés KMS de chiffrement symétriques avec des éléments de clé générés par AWS KMS, les colonnes Alias, Key ID (ID de clé), Status (État) et Creation date (Date de création) sont susceptibles d'être les plus utiles.

Clés KMS asymétriques

Si vous utilisez des clés KMS asymétriques, en plus des colonnes Alias, Key ID (ID de clé) et Status (État), envisagez d'ajouter les colonnes Key type (Type de clé), Key spec (Spécifications de la clé) et Key usage (Utilisation de la clé). Ces colonnes indiquent si une clé KMS est symétrique ou asymétrique, le type d'élément de clé et si la clé KMS peut être utilisée pour le chiffrement ou la signature.

Clés KMS HMAC

Si vous utilisez des clés KMS HMAC, en plus des colonnes Alias, Key ID (ID de clé) et Status (État), envisagez d'ajouter les colonnes Key spec (Spécifications de la clé) et Key usage (Utilisation de la clé). Ces colonnes vous indiqueront si une clé KMS est une clé HMAC. Comme vous ne pouvez pas trier les clés KMS par spécification de clé ou par utilisation des clés, utilisez des alias et des balises pour identifier vos clés HMAC, puis utilisez les fonctionnalités de filtrage de la AWS KMS console pour filtrer par alias ou balises.

Éléments de clé importés

Si vous avez des clés KMS avec des éléments de clé importés, envisagez d'ajouter les colonnes Origin (Origine) et Expiration date (Date d'expiration). Ces colonnes vous indiqueront si le contenu clé d'une clé KMS est importé ou généré par AWS KMS et quand le contenu clé expire, le cas échéant. Le champ Creation date (Date de création) affiche la date à laquelle la clé KMS a été créée (sans élément de clé). Il ne reflète aucune caractéristique de l'élément de clé.

Clés dans des magasins de clés personnalisés

Si vous avez des clés KMS dans des magasins de clés personnalisés, envisagez d'ajouter les colonnes Origin (Origine) et Custom key store ID (ID du magasin de clés personnalisé). Ces colonnes indiquent que la clé KMS se trouve dans un magasin de clés personnalisé, identifient celui-ci et affichent son type.

Clés multi-région

Si vous disposez de clés multi-région, envisagez d'ajouter la colonne Regionality (Régionalité). Cela indique si une clé KMS est une clé de région unique, une clé primaire multi-région ou une clé de réplica multi-région.

Trouvez les clés KMS et le matériel clé dans un magasin de AWS CloudHSM clés

Si vous gérez un magasin de AWS CloudHSM clés, vous devrez peut-être identifier les clés KMS dans chaque magasin de AWS CloudHSM clés. Par exemple, il se peut que vous ayez besoin de faire certaines tâches suivantes.

- Suivez les clés KMS dans le magasin de AWS CloudHSM clés dans AWS CloudTrail les journaux.
- Prédisez l'effet sur les clés KMS de la déconnexion d'un magasin de AWS CloudHSM clés.
- Planifiez la suppression des clés KMS avant de supprimer un magasin de AWS CloudHSM clés.

En outre, vous souhaiterez peut-être identifier les clés de votre AWS CloudHSM cluster qui servent de matériau clé pour vos clés KMS. Bien AWS KMS que vous gériez les clés KMS et le matériel clé, vous conservez le contrôle et la responsabilité de la gestion de votre AWS CloudHSM cluster, ainsi que des sauvegardes HSMs et des clés contenues dans le HSMs. Vous devrez peut-être identifier les clés afin d'auditer le contenu clé, de le protéger contre toute suppression accidentelle ou de le supprimer HSMs des sauvegardes de clusters après avoir supprimé la clé KMS.

Tous les éléments clés des clés KMS de votre magasin de AWS CloudHSM clés appartiennent à l'utilisateur `kmsuser` cryptographique (CU). AWS KMS définit l'attribut key label, qui n'est visible que dans AWS CloudHSM, sur le nom de ressource Amazon (ARN) de la clé KMS.

Pour rechercher les clés KMS et les éléments de clé, utilisez l'une des techniques suivantes.

- Trouvez les clés KMS dans un magasin de AWS CloudHSM clés— Comment identifier les clés KMS dans l'un ou l'ensemble de vos magasins de AWS CloudHSM clés.
- Trouvez toutes les clés d'un magasin de AWS CloudHSM clés : comment trouver toutes les clés de votre cluster qui font office d'éléments de clé pour les clés KMS dans votre magasin de clés AWS CloudHSM .
- Trouvez la AWS CloudHSM clé d'une clé KMS— Comment trouver la clé de votre cluster qui sert de matériau clé pour une clé KMS particulière dans votre magasin de AWS CloudHSM clés.
- Trouvez la clé KMS pour une AWS CloudHSM clé — Comment trouver la clé KMS pour une clé particulière de votre cluster.

Trouvez les clés KMS dans un magasin de AWS CloudHSM clés

Si vous gérez un magasin de AWS CloudHSM clés, vous devrez peut-être identifier les clés KMS dans chaque magasin de AWS CloudHSM clés. Vous pouvez utiliser ces informations pour suivre les opérations relatives aux clés KMS dans AWS CloudTrail les journaux, prévoir l'effet de la déconnexion d'un magasin de clés personnalisé sur les clés KMS ou planifier la suppression des clés KMS avant de supprimer un magasin de AWS CloudHSM clés.

Pour trouver les clés KMS dans un magasin de AWS CloudHSM clés (console)

Pour trouver les clés KMS dans un magasin de AWS CloudHSM clés en particulier, sur la page des clés gérées par le client, consultez les valeurs des champs Nom du magasin de clés personnalisé ou ID du magasin de clés personnalisé. Pour identifier les clés KMS dans n'importe quel magasin de AWS CloudHSM clés, recherchez les clés KMS dont la valeur d'origine est de AWS CloudHSM. Pour ajouter des colonnes facultatives à l'affichage, choisissez l'icône d'engrenage dans le coin supérieur droit de la page.

Pour trouver les clés KMS dans un magasin de AWS CloudHSM clés (API)

Pour rechercher les clés KMS dans un magasin de AWS CloudHSM clés, utilisez les DescribeKeyopérations ListKeyset filtrez par `CustomKeyStoreId` valeur. Avant d'exécuter les exemples suivants, remplacez les valeurs fictives de l'identifiant du magasin de clés personnalisé par une valeur valide.

Bash

Pour trouver des clés KMS dans un magasin de AWS CloudHSM clés en particulier, procurez-vous toutes vos clés KMS dans le compte et dans la région. Ensuite, filtrez sur l'ID de magasin de clés personnalisé.

```
for key in $(aws kms list-keys --query 'Keys[*].KeyId' --output text) ;
do aws kms describe-key --key-id $key |
grep '"CustomKeyStoreId": "cks-1234567890abcdef0"' --context 100; done
```

Pour obtenir des clés KMS dans n'importe quel magasin de AWS CloudHSM clés du compte et de la région, recherchez `CustomKeyStoreType` avec une valeur de`AWS_CloudHSM`.

```
for key in $(aws kms list-keys --query 'Keys[*].KeyId' --output text) ;
do aws kms describe-key --key-id $key |
grep '"CustomKeyStoreType": "AWS_CloudHSM"' --context 100; done
```

PowerShell

Pour rechercher des clés KMS dans un magasin de AWS CloudHSM clés en particulier, utilisez les KmsKey applets de commande KmsKeyList Get - pour obtenir toutes vos clés KMS dans le compte et la région. Ensuite, filtrez sur l'ID de magasin de clés personnalisé.

```
PS C:\> Get-KMSKeyList | Get-KMSKey | where CustomKeyStoreId -eq
 'cks-1234567890abcdef0'
```

Pour obtenir des clés KMS dans n'importe quel magasin de AWS CloudHSM clés du compte et de la région, filtrez en fonction de la CustomKeyStoreType valeur de`AWS_CLOUDHSM`.

```
PS C:\> Get-KMSKeyList | Get-KMSKey | where CustomKeyStoreType -eq 'AWS_CLOUDHSM'
```

Trouvez toutes les clés d'un magasin de AWS CloudHSM clés

Vous pouvez identifier les clés de votre AWS CloudHSM cluster qui servent de matériau clé pour votre magasin de AWS CloudHSM clés. Pour ce faire, utilisez la commande key list dans la CLI CloudHSM.

Vous pouvez également utiliser la commande de liste de touches pour rechercher la AWS CloudHSM touche « AWS KMS for an ». Lorsque vous AWS KMS créez le matériel clé pour une clé KMS dans

votre AWS CloudHSM cluster, il écrit le nom de ressource Amazon (ARN) de la clé KMS dans le libellé de la clé. La commande key list renvoie le `key-reference` et le`label`.

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

Pour exécuter cette procédure, vous devez déconnecter temporairement le magasin de AWS CloudHSM clés afin de pouvoir vous connecter en tant que `kmsuser` CU.

1. Déconnectez le magasin de AWS CloudHSM clés, s'il n'est pas déjà déconnecté, puis connectez-vous en tant que`kmsuser`, comme expliqué dansComment se déconnecter et se connecter.

   > **Note**
   >
   > Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

2. Utilisez la commande key list de la CLI CloudHSM pour rechercher toutes les clés de l'utilisateur actuel présent AWS CloudHSM dans votre cluster.

   Par défaut, seules 10 touches de l'utilisateur actuellement connecté sont affichées, et seules les touches `key-reference` et `label` sont affichées en sortie. Pour plus d'options, consultez la liste des clés dans le guide de AWS CloudHSM l'utilisateur.

   ```
   aws-cloudhsm > key list
   {
     "error_code": 0,
   ```

```
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000000123",
        "attributes": {
          "label": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
      },
      {
        "key-reference": "0x0000000000000456",
        "attributes": {
          "label": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
      },.
      ...8 keys later...
    ],
    "total_key_count": 56,
    "returned_key_count": 10,
    "next_token": "10"
  }
}
```

3. Déconnectez-vous et reconnectez le magasin de AWS CloudHSM clés comme décrit dansComment se déconnecter et se reconnecter.

Trouvez la clé KMS pour une AWS CloudHSM clé

Si vous connaissez la référence clé ou l'ID d'une clé qu'il `kmsuser` possède dans le cluster, vous pouvez utiliser cette valeur pour identifier la clé KMS associée dans votre magasin de AWS CloudHSM clés.

Lorsque vous AWS KMS créez le matériel clé pour une clé KMS dans votre AWS CloudHSM cluster, il écrit le nom de ressource Amazon (ARN) de la clé KMS dans le libellé de la clé. À moins que vous n'ayez modifié la valeur de l'étiquette, vous pouvez utiliser la commande key list de la CLI CloudHSM pour identifier la clé KMS associée AWS CloudHSM à la clé.

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

Pour exécuter ces procédures, vous devez déconnecter temporairement le magasin de AWS CloudHSM clés afin de pouvoir vous connecter en tant que `kmsuser` CU.

> **Note**
>
> Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

Rubriques

- Identifier la clé KMS associée à une référence clé
- Identifiez la clé KMS associée à un ID de clé de sauvegarde

Identifier la clé KMS associée à une référence clé

Les procédures suivantes montrent comment utiliser la commande key list dans la CLI CloudHSM `key-reference` avec le filtre d'attributs pour trouver la clé de votre cluster qui sert de clé pour une clé KMS spécifique AWS CloudHSM dans votre magasin de clés.

1. Déconnectez le magasin de AWS CloudHSM clés, s'il n'est pas déjà déconnecté, puis connectez-vous en tant que`kmsuser`, comme expliqué dansComment se déconnecter et se connecter.

2. Utilisez la commande key list de la CLI CloudHSM pour filtrer `key-reference` en fonction de l'attribut. Spécifiez l'`verbose`argument pour inclure tous les attributs et les informations clés de la clé correspondante. Si vous ne spécifiez pas l'`verbose`argument, l'opération de liste de clés renvoie uniquement la référence clé et l'attribut label de la clé correspondante.

   Avant d'exécuter cette commande, remplacez l'exemple `key-reference` par un exemple valide provenant de votre compte.

```
aws-cloudhsm > key list --filter attr.key-reference="0x0000000000120034" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000120034",
        "key-info": {
          "key-owners": [
            {
              "username": "kmsuser",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "id": "0xbacking-key-id",
          "check-value": "0x29bbd1",
          "class": "my_test_key",
          "encrypt": true,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": false,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
```

```
          "sensitive": true,
          "sign": false,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": true,
          "wrap-with-trusted": false,
          "key-length-bytes": 32
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Déconnectez-vous et reconnectez le magasin de AWS CloudHSM clés comme décrit dansComment se déconnecter et se reconnecter.

Identifiez la clé KMS associée à un ID de clé de sauvegarde

Toutes les entrées de CloudTrail journal relatives aux opérations cryptographiques effectuées avec une clé KMS dans un magasin de AWS CloudHSM clés incluent un `additionalEventData` champ avec le `customKeyStoreId` et`backingKeyId`. La valeur renvoyée dans le `backingKeyId` champ est en corrélation avec l'attribut clé `id` CloudHSM. Vous pouvez filtrer l'opération de liste de clés par `id` attribut afin d'identifier la clé KMS associée à une clé spécifique`backingKeyId`.

1. Déconnectez le magasin de AWS CloudHSM clés, s'il n'est pas déjà déconnecté, puis connectez-vous en tant que`kmsuser`, comme expliqué dansComment se déconnecter et se connecter.
2. Utilisez la commande key list de la CLI CloudHSM avec le filtre d'attributs pour trouver la clé de votre cluster qui sert de matériau clé pour une clé KMS spécifique AWS CloudHSM dans votre magasin de clés.

   L'exemple suivant montre comment filtrer en fonction de l'`id`attribut. AWS CloudHSM reconnaît la `id` valeur sous forme de valeur hexadécimale. Pour filtrer l'opération de liste de clés par `id` attribut, vous devez d'abord convertir la `backingKeyId` valeur que vous avez identifiée dans votre entrée de CloudTrail journal dans un format AWS CloudHSM reconnu.

a. Utilisez la commande Linux suivante pour convertir le `backingKeyId` en une représentation hexadécimale.

```
echo backingKeyId | tr -d '\n' |  xxd -p
```

L'exemple suivant montre comment convertir le tableau d'`backingKeyId`octets en une représentation hexadécimale.

```
echo 5890723622dc15f699aa9ab2387a9f744b2b884c18b2186ee8ada4f556a2eb9d | tr -d
 '\n' |  xxd -p
353839303732333632326463313566363939616139616232333837613966373434623262383834633138623
```

b. Ajoutez la représentation hexadécimale du `backingKeyId` with`0x`.

```
0x353839303732333632326463313566363939616139616232333837613966373434623262383834633138623
```

c. Utilisez la `backingKeyId` valeur convertie pour filtrer en fonction de l'`id`attribut. Spécifiez l'`verbose`argument pour inclure tous les attributs et les informations clés de la clé correspondante. Si vous ne spécifiez pas l'`verbose`argument, l'opération de liste de clés renvoie uniquement la référence clé et l'attribut label de la clé correspondante.

```
aws-cloudhsm > key list --filter
 attr.id="0x353839303732333632326463313566363939616139616232333837613966373434623262383
 --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000120034",
        "key-info": {
          "key-owners": [
            {
              "username": "kmsuser",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "full"
        },
```

```
        "attributes": {
          "key-type": "aes",
          "label": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "id":
 "0x353839303732333632326463313566363939616139616232333837613966373434623262383834633133
          "check-value": "0x29bbd1",
          "class": "my_test_key",
          "encrypt": true,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": false,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": false,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": true,
          "wrap-with-trusted": false,
          "key-length-bytes": 32
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Déconnectez-vous et reconnectez le magasin de AWS CloudHSM clés comme décrit dansComment se déconnecter et se reconnecter.

Trouvez la AWS CloudHSM clé d'une clé KMS

Vous pouvez utiliser l'ID de clé KMS d'une clé KMS dans un magasin de AWS CloudHSM clés pour identifier la clé de votre AWS CloudHSM cluster qui lui sert de matériau clé.

Lorsque vous AWS KMS créez le matériel clé pour une clé KMS dans votre AWS CloudHSM cluster, il écrit le nom de ressource Amazon (ARN) de la clé KMS dans le libellé de la clé. À moins que vous n'ayez modifié la valeur de l'étiquette, vous pouvez utiliser la commande key list de la CLI CloudHSM pour trouver la ressource clé et l'identifiant du matériau clé pour la clé KMS.

Toutes les entrées du CloudTrail journal relatives aux opérations cryptographiques effectuées avec une clé KMS dans un magasin de AWS CloudHSM clés incluent un `additionalEventData` champ avec le `customKeyStoreId` et`backingKeyId`. La valeur renvoyée dans le `backingKeyId` champ est l'attribut `id` AWS CloudHSM clé. Vous pouvez filtrer l'opération de la AWS CloudHSM CLI de liste de clés par ARN de clé KMS afin d'identifier l'attribut `id` clé CloudHSM associé à une clé KMS spécifique.

Pour exécuter cette procédure, vous devez déconnecter temporairement le magasin de AWS CloudHSM clés afin de pouvoir vous connecter en tant que `kmsuser` CU.

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

1. Déconnectez le magasin de AWS CloudHSM clés, s'il n'est pas déjà déconnecté, puis connectez-vous en tant que`kmsuser`, comme expliqué dansComment se déconnecter et se connecter.

   > **Note**
   >
   > Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

2. Utilisez la commande key list dans la CLI CloudHSM et `label` filtrez par pour trouver la clé KMS correspondant à une clé AWS CloudHSM spécifique dans votre cluster. Spécifiez l'`verbose`argument pour inclure tous les attributs et les informations clés de la clé correspondante. Si vous ne spécifiez pas l'`verbose`argument, l'opération de liste de clés renvoie uniquement les attributs de référence et d'étiquette de la clé correspondante.

   L'exemple suivant montre comment filtrer en fonction de l'`label`attribut qui stocke l'ARN de la clé KMS. Avant d'exécuter cette commande, remplacez l'exemple d'ARN de la clé KMS par une clé valide provenant de votre compte.

```
aws-cloudhsm > key list --filter attr.label="arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000120034",
        "key-info": {
          "key-owners": [
            {
              "username": "kmsuser",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "id": "0xbacking-key-id",
          "check-value": "0x29bbd1",
          "class": "my_test_key",
          "encrypt": true,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": false,
```

```
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": false,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": true,
          "wrap-with-trusted": false,
          "key-length-bytes": 32
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Déconnectez-vous et reconnectez le magasin de AWS CloudHSM clés comme décrit dansComment se déconnecter et se reconnecter.

Activer et désactiver les touches

Vous pouvez activer et désactiver les clés gérées par les clients. Lorsque vous créez une clé KMS, elle est activée par défaut. Si vous désactivez une clé KMS, elle ne peut être utilisée dans aucune opération de chiffrement jusqu'à sa réactivation.

Étant donné qu'il s'agit d'une action temporaire et réversible facilement, la désactivation d'une clé KMS constitue une alternative sûre à sa suppression, action destructrice et irréversible. Si vous envisagez de supprimer une clé KMS, désactivez-la d'abord et configurez une CloudWatch alarme ou un mécanisme similaire pour être certain de ne jamais avoir à utiliser la clé pour déchiffrer des données chiffrées.

Lorsque vous désactivez une clé KMS, elle devient immédiatement inutilisable (sous réserve d'une éventuelle cohérence). Toutefois, les ressources chiffrées à l'aide de clés de données protégées par la clé KMS ne sont pas affectées tant que la clé KMS n'est pas réutilisée, par exemple pour déchiffrer la clé de données. Ce problème concerne la Services AWS plupart d'entre eux qui utilisent des clés de données pour protéger vos ressources. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

Vous ne pouvez pas activer ou désactiver Clés gérées par AWSou Clés détenues par AWS. Clés gérées par AWS sont activés en permanence pour être utilisés par les services qui utilisent AWS KMS. Clés détenues par AWS sont gérés uniquement par le service qui les détient.

> **Note**
>
> AWS KMS ne fait pas pivoter le contenu clé des clés gérées par le client lorsqu'elles sont désactivées. Pour de plus amples informations, veuillez consulter Comment fonctionne la rotation des clés.

Utilisation de la AWS KMS console

Vous pouvez utiliser la AWS KMS console pour activer et désactiver les clés gérées par le client.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.

3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Cochez les cases en regard des clés KMS que vous voulez activer ou désactiver.
5. Pour activer une clé KMS, choisissez Key actions (Actions de clé), Enable (Activer). Pour désactiver une clé KMS, choisissez Key actions (Actions de clé), Disable (Désactiver).

Utilisation de l' AWS KMS API

L'EnableKeyopération active une personne désactivée AWS KMS key. Ces exemples utilisent l'AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge. Le paramètre `key-id` est obligatoire.

Cette opération ne renvoie aucune sortie. Pour voir l'état de la clé, utilisez l'DescribeKeyopération.

```
$ aws kms enable-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

L'DisableKeyopération désactive une clé KMS activée. Le paramètre `key-id` est obligatoire.

```
$ aws kms disable-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette opération ne renvoie aucune sortie. Pour voir l'état de la clé, utilisez l'DescribeKeyopération et consultez le `Enabled` champ.

```
$ aws kms describe-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyMetadata": {
        "Origin": "AWS_KMS",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Description": "",
        "KeyManager": "CUSTOMER",
        "MultiRegion": false,
        "Enabled": false,
        "KeyState": "Disabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "CreationDate": 1502910355.475,
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333"
        "KeySpec": "SYMMETRIC_DEFAULT",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
```

```
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

Rotation AWS KMS keys

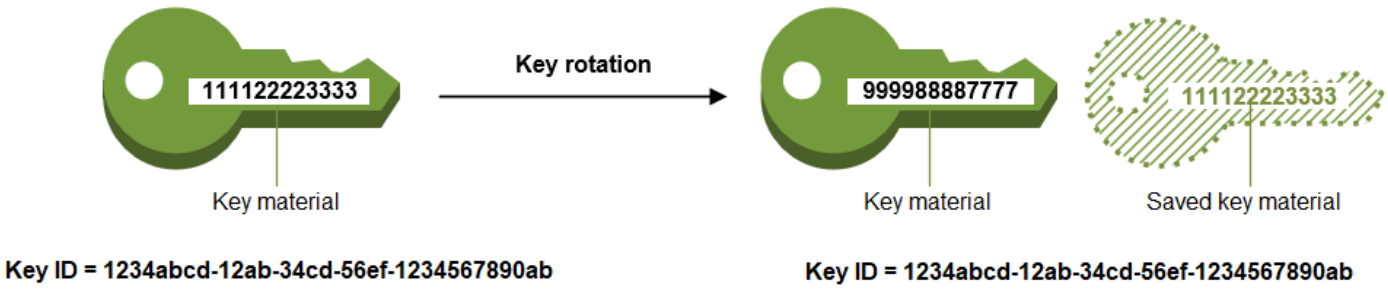
Pour créer de nouveaux éléments de chiffrement pour vos clés gérées par le client, vous pouvez créer de nouvelles clés KMS, puis modifier vos applications ou alias pour utiliser les nouvelles clés KMS. Vous pouvez également faire pivoter le matériel clé associé à une clé KMS existante en activant la rotation automatique des touches ou en effectuant une rotation à la demande.

Par défaut, lorsque vous activez la rotation automatique des clés pour une clé KMS, de nouveaux éléments cryptographiques sont AWS KMS générés chaque année pour la clé KMS. Vous pouvez également définir une option personnalisée rotation-period pour définir le nombre de jours après l'activation de la rotation automatique des clés qui AWS KMS fera pivoter votre matériel clé, ainsi que le nombre de jours entre chaque rotation automatique par la suite. Si vous devez lancer immédiatement la rotation des matériaux clés, vous pouvez effectuer une rotation à la demande, que la rotation automatique des clés soit activée ou non. Les rotations à la demande ne modifient pas les programmes de rotation automatiques existants.

Vous pouvez suivre la rotation du contenu clé de vos clés KMS sur Amazon CloudWatch et sur la AWS Key Management Service console. AWS CloudTrail Vous pouvez également utiliser le GetKeyRotationStatusmode fonctionnement pour vérifier si la rotation automatique est activée pour une clé KMS et identifier les rotations à la demande en cours. Vous pouvez utiliser ListKeyRotationsl'opération pour afficher les détails des rotations terminées.

La rotation des clés modifie uniquement le contenu de la clé actuelle, à savoir le secret cryptographique utilisé dans les opérations de chiffrement. Lorsque vous utilisez la clé KMS pivotée pour déchiffrer du texte chiffré, elle AWS KMS utilise le matériel clé utilisé pour le chiffrer. Vous ne pouvez pas sélectionner un matériau clé particulier pour les opérations de déchiffrement, il choisit AWS KMS automatiquement le bon matériau clé. Comme le déchiffre de AWS KMS manière transparente avec le matériel clé approprié, vous pouvez utiliser une clé KMS pivotée en toute sécurité dans les applications et sans modifier le code. Services AWS

La clé KMS est la même ressource logique, indépendamment du fait que ses éléments de clé changent ou du nombre de fois où ils changent. Les propriétés de la clé KMS ne changent pas, comme illustré dans l'image suivante.



Vous pouvez décider de créer une nouvelle clé KMS et de l'utiliser à la place de la clé KMS d'origine. L'effet est le même que celui obtenu par la rotation des éléments de clé dans une clé KMS existante. Ainsi, on parle souvent à ce sujet de rotation manuelle de la clé. La rotation manuelle est un bon choix lorsque vous souhaitez faire pivoter des clés KMS qui ne sont pas éligibles à la rotation automatique ou à la demande, notamment les clés KMS asymétriques, les clés HMAC KMS, les clés KMS dans des magasins de clés personnalisés et les clés KMS multirégionales dont le contenu clé est importé.

> **Note**
>
> La rotation des clés n'a aucun effet sur les données protégées par la clé KMS. Il ne fait pas pivoter les clés de données générées par la clé KMS et ne chiffre pas à nouveau les données protégées par la clé KMS. La rotation des clés n'atténuera pas l'effet d'une clé de données compromise.

Rotation des clés et tarification

AWS KMS facture des frais mensuels pour la première et la deuxième rotation du matériel clé conservé pour votre clé KMS. Cette augmentation de prix est plafonnée lors de la deuxième rotation, et les rotations suivantes ne seront pas facturées. Pour plus d'informations, consultez Tarification AWS Key Management Service.

> **Note**
>
> Vous pouvez utiliser le AWS Cost Explorer Service pour consulter les détails de vos frais de stockage de clés. Par exemple, vous pouvez filtrer votre affichage pour voir le montant total des frais pour les clés facturées en tant que clés KMS actuelles ou ayant fait l'objet d'une rotation en spécifiant `$REGION-KMS-Keys` pour le type d'utilisation et en regroupant les données par opération d'API.

Vous pouvez toujours voir des instances de l'ancienne opération d'API Unknown pour les dates historiques.

Rotation des clés et quotas

Chaque clé KMS compte comme une clé lors du calcul des quotas de ressources de clés, quel que soit le nombre de versions d'éléments de clé qui ont fait l'objet d'une rotation.

Pour plus d'informations sur les éléments de clé et la rotation, veuillez consulter le livre blanc Détails cryptographiques de AWS Key Management Service.

Rubriques

Pourquoi faire pivoter les clés KMS ?

Les meilleures pratiques cryptographiques découragent la réutilisation intensive des clés qui chiffrent directement les données, telles que les clés de données générées. AWS KMS Lorsque des clés de données à 256 bits chiffrent des millions de messages, elles peuvent s'épuiser et commencer à produire du texte chiffré avec des motifs subtils que des acteurs intelligents peuvent exploiter pour découvrir les bits contenus dans la clé. Il est préférable d'utiliser les clés de données une seule fois, ou seulement quelques fois, pour éviter cet épuisement des clés.

Cependant, les clés KMS sont le plus souvent utilisées comme clés d'encapsulage, également appelées clés de chiffrement. Au lieu de chiffrer les données, les clés d'encapsulage chiffrent les clés de données qui chiffrent vos données. Elles sont donc beaucoup moins souvent utilisées que les clés de données et ne sont presque jamais suffisamment réutilisées pour risquer d'épuiser les clés.

Malgré ce très faible risque d'épuisement, vous devrez peut-être alterner vos clés KMS en raison de règles commerciales ou contractuelles ou de réglementations gouvernementales. Lorsque vous êtes obligé de faire pivoter les touches KMS, nous vous recommandons d'utiliser la rotation automatique des touches là où elle est prise en charge, d'utiliser la rotation à la demande si la rotation automatique n'est pas prise en charge, et la rotation manuelle des touches lorsque ni la rotation automatique ni la rotation à la demande ne sont prises en charge.

Vous pouvez envisager d'effectuer des rotations à la demande pour démontrer les principales fonctionnalités de rotation des matériaux ou pour valider des scripts d'automatisation. Nous recommandons d'utiliser des rotations à la demande pour les rotations imprévues et d'utiliser la rotation automatique des clés avec une période de rotation personnalisée dans la mesure du possible.

Comment fonctionne la rotation des clés

La rotation des touches AWS KMS est conçue pour être transparente et facile à utiliser. AWS KMS prend en charge la rotation automatique et à la demande optionnelle des clés uniquement pour les clés gérées par le client.

Rotation automatique des touches

AWS KMS fait automatiquement pivoter la clé KMS à la prochaine date de rotation définie par votre période de rotation. Vous n'avez pas besoin de vous souvenir de la mise à jour ou de la planifier.

La rotation automatique des clés n'est prise en charge que sur les clés KMS de chiffrement symétriques dont le contenu clé est AWS KMS généré (`AWS_KMS`origine).

La rotation automatique est facultative pour les clés KMS gérées par le client. AWS KMS fait toujours alterner le matériel clé pour les clés KMS AWS gérées chaque année. La rotation des clés KMS AWS détenues est gérée par Service AWS le propriétaire de la clé.

Rotation à la demande

Lancez immédiatement la rotation du matériel clé associé à votre clé KMS, que la rotation automatique des clés soit activée ou non.

La rotation des clés à la demande est prise en charge sur les clés KMS de chiffrement symétriques avec le matériel clé qui AWS KMS génère (`AWS_KMS`origine) et sur les clés KMS de chiffrement symétrique à région unique avec le matériel clé importé (`EXTERNAL`origine).

Rotation manuelle

Ni la rotation automatique ni la rotation des clés à la demande n'est prise en charge pour les types de clés KMS suivants, mais vous pouvez faire pivoter ces clés KMS manuellement.

- Clés KMS asymétriques
- Clés KMS HMAC
- Clés KMS dans des magasins de clés personnalisés
- Clés KMS multirégionales avec matériel clé importé

Gestion des éléments de clé

AWS KMS conserve tous les éléments clés d'une clé KMS avec `AWS_KMS` origine, même si la rotation des clés est désactivée. AWS KMS supprime le contenu clé uniquement lorsque vous supprimez la clé KMS.

Vous gérez les éléments clés des clés de chiffrement symétriques avec `EXTERNAL` origine. Vous pouvez supprimer n'importe quel matériau clé à l'aide de cette DeleteImportedKeyMaterialopération ou définir une date d'expiration lors de l'importation du matériau. La clé KMS devient inutilisable dès que l'un de ses éléments expire ou est supprimé.

Utilisation des éléments de clé

Lorsque vous utilisez une clé KMS pivotée pour chiffrer des données, AWS KMS utilise le contenu de la clé actuelle. Lorsque vous utilisez la clé KMS qui a fait l'objet d'une rotation pour déchiffrer le texte chiffré, AWS KMS utilise la même version des éléments de clé qui a été utilisée pour les chiffrer. Vous ne pouvez pas sélectionner une version particulière du matériel clé pour les opérations de déchiffrement, vous choisissez AWS KMS automatiquement la bonne version.

Période de rotation

La période de rotation définit le nombre de jours après l'activation de la rotation automatique des clés qui AWS KMS fera pivoter votre matériel clé, ainsi que le nombre de jours entre chaque rotation automatique des clés par la suite. Si vous ne spécifiez aucune valeur pour `RotationPeriodInDays` activer la rotation automatique des touches, la valeur par défaut est de 365 jours.

Vous pouvez utiliser la clé de RotationPeriodInDays condition kms : pour restreindre davantage les valeurs que les principaux peuvent spécifier dans le `RotationPeriodInDays` paramètre.

Date de rotation

La date de rotation reflète la date à laquelle le contenu clé actuel d'une clé KMS a été mis à jour à la suite d'une rotation automatique (planifiée) ou d'une rotation de clé à la demande.

Date de rotation

AWS KMS fait automatiquement pivoter la clé KMS à la date de rotation définie par votre période de rotation. La période de rotation par défaut est de 365 jours.

Clés gérées par le client

La rotation automatique des clés étant facultative sur les clés gérées par le client et pouvant être activée ou désactivée à tout moment, la date de rotation dépend de la date à laquelle la rotation a été activée pour la dernière fois. La date peut changer si vous modifiez la période de rotation d'une clé pour laquelle vous avez précédemment activé la rotation automatique des touches. La date de rotation peut changer plusieurs fois au cours de la durée de vie de la clé.

Par exemple, si vous créez une clé gérée par le client le 1er janvier 2022 et que vous activez la rotation automatique des clés avec une période de rotation par défaut de 365 jours le 15 mars 2022, vous faites AWS KMS pivoter le contenu clé le 15 mars 2023, le 15 mars 2024, puis tous les 365 jours par la suite.

Les exemples suivants supposent que la rotation automatique des clés a été activée avec une période de rotation par défaut de 365 jours. Ces exemples illustrent des cas particuliers susceptibles d'avoir un impact sur la période de rotation d'une clé.

- Désactiver la rotation des clés : si vous désactivez la rotation automatique des clés à tout moment, la clé KMS continue d'utiliser la version de l'élément de clé qu'elle utilisait lorsque la rotation a été désactivée. Si vous réactivez la rotation automatique des touches, AWS KMS fait pivoter le matériau clé en fonction de la nouvelle date d'activation de la rotation.
- Clés KMS désactivées : lorsqu'une clé KMS est désactivée, elle AWS KMS ne fait pas pivoter. Toutefois, l'état de rotation de la clé ne change pas et vous ne pouvez pas le modifier tant que la clé KMS est désactivée. Lorsque la clé KMS est réactivée, si le contenu clé a dépassé sa dernière date de rotation planifiée, il la AWS KMS fait immédiatement pivoter. Si le matériel clé n'a pas dépassé sa dernière date de rotation planifiée, AWS KMS reprend le calendrier de rotation des clés d'origine.
- Clés KMS en attente de suppression — Lorsqu'une clé KMS est en attente de suppression, elle AWS KMS ne fait pas l'objet d'une rotation. L'état de rotation de la clé est défini sur `false` et vous ne pouvez pas le modifier tant que la suppression est en attente. Si la

suppression est annulée, l'état précédent de rotation de la clé est restauré. Si le matériau clé a dépassé sa dernière date de rotation planifiée, il le AWS KMS fait immédiatement pivoter. Si le matériel clé n'a pas dépassé sa dernière date de rotation planifiée, AWS KMS reprend le calendrier de rotation des clés d'origine.

Clés gérées par AWS

AWS KMS effectue une rotation automatique Clés gérées par AWS chaque année (environ 365 jours). Vous ne pouvez pas activer ou désactiver la rotation des clés pour Clés gérées par AWS.

Le matériau clé d'un Clé gérée par AWS est d'abord alterné un an après sa date de création, puis chaque année (environ 365 jours après la dernière rotation) par la suite.

> **Note**
>
> En mai 2022, le calendrier de rotation AWS KMS a été modifié, Clés gérées par AWS passant de tous les trois ans (environ 1 095 jours) à chaque année (environ 365 jours).

Clés détenues par AWS

Vous ne pouvez pas activer ou désactiver la rotation des clés pour Clés détenues par AWS. La stratégie de rotation des clés pour un Clé détenue par AWS est déterminée par le AWS service qui crée et gère la clé. Pour plus de détails, reportez-vous à la rubrique Chiffrement au repos dans le Guide de l'utilisateur ou le guide du développeur du service.

Touches multirégionales rotatives

Vous pouvez activer et désactiver la rotation automatique et effectuer une rotation à la demande du contenu clé dans le chiffrement symétrique. Clés multirégionales avec `AWS_KMS` origine. La rotation des clés est une propriété partagée des clés multirégionales.

Vous activez et désactivez la rotation automatique des clés uniquement sur la clé principale. Vous initiez la rotation à la demande uniquement sur la clé primaire.

- Lors de la AWS KMS synchronisation des clés multirégionales, il copie le paramètre de propriété de rotation des clés de la clé primaire vers toutes les clés répliques associées.
- Lorsqu'il AWS KMS fait pivoter le matériau clé, il crée un nouveau matériau clé pour la clé primaire, puis copie le nouveau matériau clé au-delà des limites de la région vers toutes les répliques de clés associées. Le contenu clé ne sort jamais AWS KMS non chiffré. Cette

étape est soigneusement contrôlée pour s'assurer que les éléments de clé sont entièrement synchronisés avant qu'une clé ne soit utilisée dans une opération cryptographique.

- AWS KMS ne chiffre aucune donnée avec le nouveau matériel clé tant que ce matériel clé n'est pas disponible dans la clé primaire et dans chacune de ses clés répliques.
- Lorsque vous répliquez une clé principale qui a fait l'objet d'une rotation, la nouvelle clé de réplica possède les éléments de clé actuels et toutes les versions précédentes des éléments de clé pour ses clés multi-région associées.

Ce modèle garantit que les clés multi-région associées sont entièrement interopérables. Toute clé multi-région peut déchiffrer tout texte chiffré par une clé multi-région associée, même si le texte chiffré a été chiffré avant la création de la clé.

AWS services

Vous pouvez activer la rotation automatique des clés sur les clés gérées par le client que vous utilisez pour le chiffrement côté serveur dans les services AWS . La rotation annuelle est transparente et compatible avec les services AWS .

Surveillance de la rotation des clés

Lorsqu'il AWS KMS fait pivoter le contenu clé d'une clé Clé gérée par AWSou d'une clé gérée par le client, il écrit un `KMS CMK Rotation` événement sur Amazon EventBridge et un RotateKey autre dans votre AWS CloudTrail journal. Vous pouvez utiliser ces registres pour vérifier que la clé KMS a fait l'objet d'une rotation.

Vous pouvez utiliser la AWS Key Management Service console pour afficher le nombre de rotations à la demande restantes et une liste de toutes les rotations de matériaux clés terminées pour une clé KMS.

Vous pouvez utiliser ListKeyRotationsl'opération pour afficher les détails des rotations terminées.

Cohérence à terme

La rotation des clés est soumise aux mêmes effets de cohérence éventuels que les autres opérations AWS KMS de gestion. Il peut y avoir un léger retard avant que les nouveaux éléments de clé ne soient disponibles dans AWS KMS. Toutefois, la rotation des éléments de clé n'entraîne aucune interruption ou aucun retard dans les opérations cryptographiques. Les éléments de clé actuels sont utilisés dans les opérations cryptographiques jusqu'à ce que les nouveaux éléments de clé soient disponibles dans AWS KMS. Lorsque le matériau clé d'une clé multirégionale est automatiquement pivoté, AWS KMS utilise le matériau clé actuel jusqu'à ce que le nouveau matériau clé soit disponible dans toutes les régions avec une clé multirégionale associée.

Activer la rotation automatique des touches

Par défaut, lorsque vous activez la rotation automatique des clés pour une clé KMS, de nouveaux éléments cryptographiques sont AWS KMS générés chaque année pour la clé KMS. Vous pouvez également définir une option personnalisée rotation-period pour définir le nombre de jours après l'activation de la rotation automatique des clés qui AWS KMS fera pivoter votre matériel clé, ainsi que le nombre de jours entre chaque rotation automatique par la suite.

La rotation automatique des clés offre les avantages suivants :

- Les propriétés de la clé KMS, y compris son ID de clé, son ARN de clé, sa région, ses politiques et ses autorisations, ne changent pas lorsque la clé est l'objet d'une rotation.
- Vous n'avez pas besoin de modifier les applications ou les alias qui font référence à l'ID ou à l'ARN de la clé KMS.
- La rotation des éléments de clé n'affecte pas l'utilisation de la clé KMS dans Service AWS.
- Après avoir activé la rotation des clés, AWS KMS fait automatiquement pivoter la clé KMS à la date de rotation suivante définie par votre période de rotation. Vous n'avez pas besoin de vous souvenir de la mise à jour ou de la planifier.

Vous pouvez activer la rotation automatique des touches dans la AWS KMS console ou en utilisant l'EnableKeyRotationopération. Pour activer la rotation automatique des touches, vous devez disposer d'`kms:EnableKeyRotation`autorisations. Pour plus d'informations sur AWS KMS les autorisations, consultez leRéférence des autorisations .

Utilisation de la AWS KMS console

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas activer ou désactiver la rotation des Clés gérées par AWS. Elles sont automatiquement soumises à la rotation tous ans.)
4. Choisissez l'alias ou l'ID d'une clé KMS.
5. Choisissez l'onglet Rotation des clés d'accès.

L'onglet Rotation des clés apparaît uniquement sur la page détaillée des clés KMS de chiffrement symétriques dont le contenu a été AWS KMS généré (l'origine est AWS_KMS), y compris les clés KMS de chiffrement symétriques multirégionales.

Vous ne pouvez pas soumettre automatiquement à la rotation les clés KMS asymétriques, les clés KMS HMAC, les clés KMS avec des éléments de clé importés, ou les clés KMS dans les magasins de clé personnalisés. Cependant, vous pouvez les faire pivoter manuellement.

6. Dans la section Rotation automatique des touches, choisissez Modifier.
7. Pour Rotation des touches, sélectionnez Activer.

> **Note**
>
> Si une clé KMS est désactivée ou en attente de suppression, AWS KMS cela ne fait pas pivoter le contenu clé et vous ne pouvez pas mettre à jour l'état de rotation automatique des clés ou la période de rotation. Activez la clé KMS ou annulez la suppression pour mettre à jour la configuration de rotation automatique des clés. Pour plus d'informations, consultez Comment fonctionne la rotation des clés et États clés des AWS KMS clés.

8. (Facultatif) Entrez une période de rotation comprise entre 90 et 2560 jours. La valeur par défaut est de 365 jours. Si vous ne spécifiez pas de période de rotation personnalisée, le matériau clé AWS KMS sera alterné chaque année.

   Vous pouvez utiliser la clé de RotationPeriodInDays condition kms : pour limiter les valeurs que les directeurs peuvent spécifier pour la période de rotation.
9. Choisissez Enregistrer.

Utilisation de l' AWS KMS API

Vous pouvez utiliser l'API AWS Key Management Service (AWS KMS) pour activer la rotation automatique des clés et consulter l'état de rotation actuel de toute clé gérée par le client. Ces exemples utilisent l'AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

L'EnableKeyRotationopération active la rotation automatique des touches pour la clé KMS spécifiée. Pour identifier la clé KMS dans cette opération, utilisez son ID de clé ou son ARN de clé. Par défaut, la rotation des clés est désactivée pour les clés gérées par le client.

Vous pouvez utiliser la clé de kms:RotationPeriodInDayscondition pour limiter les valeurs que les principaux peuvent spécifier pour le `RotationPeriodInDays` paramètre d'une `EnableKeyRotation` demande.

L'exemple suivant active la rotation des clés avec une période de rotation de 180 jours sur la clé KMS de chiffrement symétrique spécifiée et utilise l'GetKeyRotationStatusopération pour voir le résultat.

```
$ aws kms enable-key-rotation \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --rotation-period-in-days 180

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyRotationEnabled": true,
    "RotationPeriodInDays": 180,
    "NextRotationDate": "2024-02-14T18:14:33.587000+00:00"
}
```

Désactiver la rotation automatique des touches

Après avoir activé la rotation automatique des clés sur une clé gérée par le client, vous pouvez choisir de la désactiver à tout moment.

Si vous désactivez la rotation automatique des touches, la clé KMS continue d'utiliser la version du matériel clé qu'elle utilisait lorsque la rotation a été désactivée. Si vous réactivez la rotation automatique des touches, AWS KMS fait pivoter le matériau clé en fonction de la nouvelle date d'activation de la rotation.

La désactivation de la rotation automatique n'a aucune incidence sur votre capacité à effectuer des rotations à la demande et n'annule aucune rotation à la demande en cours.

Vous pouvez désactiver la rotation automatique des touches dans la AWS KMS console ou en utilisant l'DisableKeyRotationopération. Pour désactiver la rotation automatique des touches, vous devez disposer d'`kms:DisableKeyRotation`autorisations. Pour plus d'informations sur AWS KMS les autorisations, consultez leRéférence des autorisations .

Utilisation de la AWS KMS console

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.

2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas activer ou désactiver la rotation des Clés gérées par AWS. Elles sont automatiquement soumises à la rotation tous ans.)
4. Choisissez l'alias ou l'ID d'une clé KMS.
5. Choisissez l'onglet Rotation des clés d'accès.

   L'onglet Rotation des clés apparaît uniquement sur la page détaillée des clés KMS de chiffrement symétriques dont le contenu a été AWS KMS généré (l'origine est AWS_KMS), y compris les clés KMS de chiffrement symétriques multirégionales.

   Vous ne pouvez pas soumettre automatiquement à la rotation les clés KMS asymétriques, les clés KMS HMAC, les clés KMS avec des éléments de clé importés, ou les clés KMS dans les magasins de clé personnalisés. Cependant, vous pouvez les faire pivoter manuellement.
6. Dans la section Rotation automatique des touches, choisissez Modifier.
7. Pour Rotation des touches, sélectionnez Désactiver.

   > **Note**
   >
   > Si une clé KMS est désactivée ou en attente de suppression, AWS KMS cela ne fait pas pivoter le contenu clé et vous ne pouvez pas mettre à jour l'état de rotation automatique des clés ou la période de rotation. Activez la clé KMS ou annulez la suppression pour mettre à jour la configuration de rotation automatique des clés. Pour plus de détails, veuillez consulter Comment fonctionne la rotation des clés et États clés des AWS KMS clés.
8. Choisissez Enregistrer.

Utilisation de l' AWS KMS API

Vous pouvez utiliser l'API AWS Key Management Service (AWS KMS) pour désactiver la rotation automatique des clés et consulter l'état de rotation actuel de toute clé gérée par le client. Cet exemple utilise le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

L'DisableKeyRotationopération désactive la rotation automatique des touches. Pour identifier la clé KMS dans cette opération, utilisez son ID de clé ou son ARN de clé. Par défaut, la rotation des clés est désactivée pour les clés gérées par le client.

L'exemple suivant désactive la rotation automatique des clés sur la clé KMS de chiffrement symétrique spécifiée et utilise l'GetKeyRotationStatusopération pour voir le résultat.

```
$ aws kms disable-key-rotation --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyRotationEnabled": false
}
```

Effectuez une rotation des touches à la demande

Vous pouvez effectuer une rotation à la demande du contenu clé des clés KMS gérées par le client, que la rotation automatique des clés soit activée ou non. La désactivation de la rotation automatique (DisableKeyRotation) n'a aucune incidence sur votre capacité à effectuer des rotations à la demande et n'annule aucune rotation à la demande en cours. Les rotations à la demande ne modifient pas les programmes de rotation automatiques existants. Prenons l'exemple d'une clé KMS dont la rotation automatique des clés est activée avec une période de rotation de 730 jours. Si la rotation de la clé est prévue automatiquement le 14 avril 2024 et que vous effectuez une rotation à la demande le 10 avril 2024, la clé tournera automatiquement, comme prévu, le 14 avril 2024 et tous les 730 jours par la suite.

Vous pouvez effectuer une rotation de clé à la demande au maximum 10 fois par clé KMS. Vous pouvez utiliser la AWS KMS console pour afficher le nombre de rotations à la demande restantes disponibles pour une clé KMS.

La rotation des clés à la demande n'est prise en charge que sur les clés KMS de chiffrement symétriques. Vous ne pouvez pas effectuer de rotation à la demande de clés KMS asymétriques, de clés KMS HMAC, de clés KMS multirégionales avec des éléments clés importés ou de clés KMS dans un magasin de clés personnalisé. Pour effectuer la rotation à la demande d'un ensemble de clés multirégionales associées, appelez la rotation à la demande sur la clé primaire.

Les utilisateurs autorisés disposant d'`kms:GetKeyRotationStatus`autorisations `kms:RotateKeyOnDemand` et d'autorisations peuvent utiliser la AWS KMS console et l' AWS KMS API pour lancer la rotation des clés à la demande et consulter l'état de la rotation des clés. ListKeyRotationsÀ utiliser pour afficher les rotations terminées pour une clé KMS.

Rubriques

- Lancer la rotation des touches à la demande (console)
- Lancer la rotation des clés à la demande (AWS KMS API)

Lancer la rotation des touches à la demande (console)

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas effectuer de rotation à la demande de Clés gérées par AWS. Ils font l'objet d'une rotation automatique chaque année.)
4. Choisissez l'alias ou l'ID d'une clé KMS.
5. Choisissez l'onglet Matériau clé et rotations.

   L'onglet Matériau clé et rotations apparaît uniquement sur la page détaillée des clés KMS de chiffrement symétrique qui prennent en charge la rotation automatique ou à la demande. Cela inclut les clés KMS avec le matériel clé AWS KMS généré (AWS_KMSorigine) et les clés KMS à région unique avec le matériel clé importé (origine EXTERNE).

   Vous ne pouvez pas effectuer de rotation à la demande de clés KMS asymétriques, de clés KMS HMAC, de clés KMS multirégionales avec des éléments clés importés ou de clés KMS dans des magasins de clés personnalisés. Cependant, vous pouvez les faire pivoter manuellement.
6. Choisissez Rotation maintenant. Pour les clés de chiffrement symétriques à région unique dont le contenu clé est importé, l'option Faire pivoter maintenant n'est disponible que si vous avez déjà importé un nouveau contenu clé et que celui-ci est dans l'état En attente de rotation.
7. Lisez et prenez en compte l'avertissement et les informations concernant le nombre de rotations à la demande restantes pour la clé. Vous verrez également des informations telles que l'identifiant, la description et le délai d'expiration du matériel clé qui deviendra à jour après la rotation. Si vous décidez de ne pas procéder à la rotation à la demande, choisissez Annuler.

8. Choisissez la touche Rotation pour confirmer la rotation à la demande.

> **Note**
>
> La rotation à la demande est soumise aux mêmes effets de cohérence éventuels que les autres opérations AWS KMS de gestion. Il peut y avoir un léger retard avant que les nouveaux éléments de clé ne soient disponibles dans AWS KMS. La bannière en haut de la console vous avertit lorsque la rotation à la demande est terminée.

Lancer la rotation des clés à la demande (AWS KMS API)

Vous pouvez utiliser l'API AWS Key Management Service (AWS KMS) pour lancer une rotation des clés à la demande et consulter l'état de rotation actuel de toute clé gérée par le client. Cet exemple utilise le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

L'RotateKeyOnDemandopération lance immédiatement une rotation de clé à la demande pour la clé KMS spécifiée. Pour identifier la clé KMS dans ces opérations, utilisez son ID de clé ou son ARN de clé.

L'exemple suivant lance une rotation de clé à la demande sur la clé KMS de chiffrement symétrique spécifiée et utilise l'GetKeyRotationStatusopération pour vérifier que la rotation à la demande est en cours. Le contenu `OnDemandRotationStartDate` de la `kms:GetKeyRotationStatus` réponse indique la date et l'heure auxquelles une rotation à la demande en cours a été initiée. Dans cet exemple, la rotation automatique de la clé KMS est également activée sur une période de 365 jours.

```
$ aws kms rotate-key-on-demand --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
}

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyRotationEnabled": true,
    "NextRotationDate": "2024-03-14T18:14:33.587000+00:00",
    "OnDemandRotationStartDate": "2024-02-24T18:44:48.587000+00:00"
    "RotationPeriodInDays": 365
}
```

Si la clé KMS ne prend pas en charge la rotation automatique ou si la rotation automatique n'est pas activée, la `kms:GetKeyRotationStatus` réponse comportera moins de champs, comme indiqué dans l'exemple suivant :

```
$ aws kms rotate-key-on-demand --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
}

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyRotationEnabled": false,
    "OnDemandRotationStartDate": "2024-02-24T18:44:48.587000+00:00"
}
```

Répertorier les rotations et les principaux matériaux

Les clés KMS qui prennent en charge la rotation automatique ou à la demande peuvent être associées à plusieurs éléments clés. Ces clés ont un matériau clé initial et un matériau clé supplémentaire pour chaque rotation automatique ou à la demande.

Les utilisateurs autorisés `kms:ListKeyRotations` autorisés peuvent utiliser la AWS KMS console et l'ListKeyRotationsAPI pour répertorier tous les éléments clés associés à une clé KMS, y compris ceux issus de rotations automatiques et à la demande effectuées.

Rubriques

- Lister les rotations et les matériaux clés (console)
- Liste des rotations et des matériaux clés (AWS KMS API)

Lister les rotations et les matériaux clés (console)

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.

4. Choisissez l'alias ou l'ID d'une clé KMS.
5. Choisissez l'onglet Matériau clé et rotations.

   - L'onglet Matériau clé et rotations apparaît uniquement sur la page détaillée des clés KMS de chiffrement symétrique qui prennent en charge la rotation automatique ou à la demande. Cela inclut les clés KMS avec le matériel clé AWS KMS généré (`AWS_KMS`origine) et les clés KMS à région unique avec le matériel clé importé (`EXTERNAL`origine).
   - Le tableau des matériaux clés de l'onglet Matériaux clés et rotations répertorie tous les matériaux clés associés à la clé KMS. Pour chaque matériau clé, l'entrée correspondante affiche son identifiant unique attribué par AWS KMS, la date de rotation et l'état du matériau clé. La date de rotation indique le moment où le contenu clé est devenu actuel après une rotation des clés automatique ou à la demande. Aucune date de rotation n'est associée au premier matériau ou au matériau `Pending rotation` clé. L'état du matériau clé détermine la manière dont le matériau clé est AWS KMS utilisé. Le matériel clé actuel est utilisé à la fois pour le chiffrement et le déchiffrement. Les éléments clés non courants ne sont utilisés que pour le déchiffrement. Un état du matériau clé de `Pending rotation` indique que le matériau clé est préparé pour la rotation. Ce matériel clé n'est utilisé pour aucune opération cryptographique jusqu'à ce qu'une rotation de clé à la demande en fasse le matériau clé actuel. Les informations supplémentaires affichées pour le matériel clé dépendent du type de clé KMS.
   - Pour les clés KMS de chiffrement symétriques avec `AWS_KMS` origine, chaque ligne affiche également le type de rotation, `On-demand` ou`Automatic`.
   - Les clés KMS de chiffrement symétrique à région unique dont le contenu clé est importé (`EXTERNAL`origine) ne prennent en charge que la `On-demand` rotation, il n'y a donc pas de colonne de type de rotation. Au lieu de cela, chaque ligne affiche un état d'importation, une description spécifiée par l'utilisateur, des informations d'expiration et un menu Actions. L'état d'importation est soit Importé, ce qui indique que le matériau clé est disponible à l'intérieur, AWS KMS soit En attente d'importation, indiquant que le matériau clé n'est pas disponible à l'intérieur AWS KMS. Le menu Actions peut être utilisé pour supprimer le matériel clé importé ou pour réimporter le matériel clé. L'action Supprimer le matériau clé est désactivée si l'état d'importation du matériau clé est En attente d'importation. L'action Réimporter le matériel clé est toujours disponible. Il n'est pas nécessaire d'attendre qu'un élément clé expire ou soit supprimé avant de le réimporter.

Liste des rotations et des matériaux clés (AWS KMS API)

Vous pouvez utiliser l'API AWS Key Management Service (AWS KMS) pour lancer une rotation des clés à la demande et consulter l'état de rotation actuel de toute clé gérée par le client. Cet exemple utilise le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

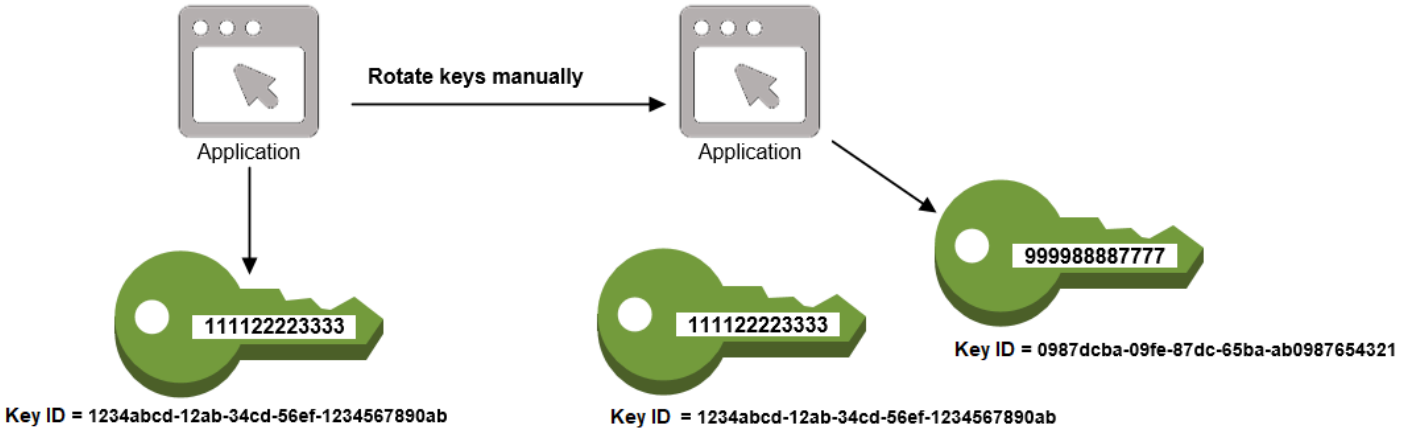
L'ListKeyRotationsopération répertorie toutes les rotations et les matériaux clés pour la clé KMS spécifiée. Pour identifier la clé KMS dans ces opérations, utilisez son ID de clé ou son ARN de clé.

Cette opération prend en charge un `IncludeKeyMaterial` paramètre facultatif. La valeur par défaut de ce paramètre est `ROTATIONS_ONLY`. Si vous omettez ce paramètre, AWS KMS renvoie des informations sur les éléments clés créés par rotation automatique ou à la demande des touches. Lorsque vous spécifiez une valeur de`ALL_KEY_MATERIAL`, AWS KMS ajoute le premier élément clé et tout élément clé importé en attente de rotation dans la réponse. Ce paramètre ne peut être utilisé qu'avec les clés KMS qui prennent en charge la rotation automatique ou à la demande des clés.

```
$ aws kms list-key-rotations --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --inlcude-key-material ALL_KEY_MATERIAL
{
    "Rotations": [
        {
            "KeyId": 1234abcd-12ab-34cd-56ef-1234567890ab,
            "KeyMaterialId":
 123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0,
            "KeyMaterialDescription": "KeyMaterialA",
            "ImportState": "PENDING_IMPORT",
            "KeyMaterialState": "NON_CURRENT"
        },
        {
            "KeyId": 1234abcd-12ab-34cd-56ef-1234567890ab,
            "KeyMaterialId":
 96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068,
            "ImportState": "IMPORTED",
            "KeyMaterialState": "CURRENT",
            "ExpirationModel": "KEY_MATERIAL_DOES_NOT_EXPIRE",
            "RotationDate": "2025-05-01T15:50:51.045000-07:00",
            "RotationType": "ON_DEMAND"
        }
    ],
    "Truncated": false
}
```

Faites pivoter les touches manuellement

Vous souhaiterez peut-être créer une nouvelle clé KMS et l'utiliser à la place d'une clé KMS actuelle au lieu d'utiliser la rotation automatique ou à la demande des clés. Lorsque la nouvelle clé KMS possède des éléments de chiffrement différents de ceux de la clé KMS actuelle, l'utilisation de la nouvelle clé KMS a le même effet que la modification des éléments de clé d'une clé KMS existante. Le processus de remplacement d'une cléKMS par une autre est connu sous le nom de rotation manuelle de clés.



La rotation manuelle est un bon choix lorsque vous souhaitez faire pivoter des clés KMS qui ne sont pas éligibles à la rotation automatique ou à la demande, telles que les clés KMS asymétriques, les clés HMAC KMS, les clés KMS dans des magasins de clés personnalisés et les clés KMS multirégionales dont le contenu clé est importé.

> **Note**
>
> Lorsque vous commencez à utiliser la nouvelle clé KMS, veillez à ce que la clé KMS d'origine reste activée afin de AWS KMS pouvoir déchiffrer les données chiffrées par la clé KMS d'origine.

Lorsque vous faites tourner les clés KMS manuellement, vous devez également mettre à jour les références à l'ID ou à l'ARN de la clé KMS dans vos applications. Les alias, qui associent un nom convivial à une clé KMS, facilitent ce processus. Utilisez un alias pour faire référence à une clé KMS dans vos applications. Ensuite, lorsque vous souhaitez modifier la clé KMS que l'application utilise,

au lieu de modifier le code de votre application, modifiez la clé KMS cible de l'alias. Pour en savoir plus, consultez Apprenez à utiliser des alias dans vos applications.

> **Note**
>
> Les alias qui pointent vers la dernière version d'une clé KMS pivotée manuellement constituent une bonne solution pour DescribeKeyles opérations cryptographiques telles que Encrypt DeriveSharedSecret,,,,, Sign GetPublicKeyGenerateDataKeyand GenerateDataKeyPairGenerateMacVerify VerifyMac. Les alias ne sont pas autorisés dans les opérations qui gèrent les clés KMS, telles que DisableKeyou ScheduleKeyDeletion.
> Lorsque vous appelez l'opération Decrypt sur des clés KMS de chiffrement symétriques pivotées manuellement, omettez le `KeyId` paramètre dans la commande. AWS KMS utilise automatiquement la clé KMS qui a chiffré le texte chiffré.
> Le `KeyId` paramètre est obligatoire lors d'un appel `Decrypt` ou d'une vérification avec une clé KMS asymétrique, ou lors d'un appel VerifyMacavec une clé KMS HMAC. Ces demandes échouent lorsque la valeur de`KeyId`est un alias qui ne pointe plus vers la clé KMS qui a effectué l'opération cryptographique, par exemple lorsqu'une clé fait l'objet d'une rotation manuelle. Pour éviter cette erreur, vous devez spécifier et suivre la clé bonne KMS pour chaque opération.

Pour modifier la clé KMS cible d'un alias, utilisez UpdateAliasl'opération dans l' AWS KMS API. Par exemple, cette commande met à jour l'alias `alias/TestKey` pour pointer vers une nouvelle clé KMS. Comme l'opération ne renvoie aucune sortie, l'exemple utilise l'ListAliasesopération pour montrer que l'alias est désormais associé à une autre clé KMS et que le `LastUpdatedDate` champ est mis à jour. Les ListAliases commandes utilisent le `query`paramètre du AWS CLI pour obtenir uniquement l'`alias/TestKey`alias.

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/TestKey`]'
{
    "Aliases": [
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/TestKey",
            "AliasName": "alias/TestKey",
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1521097200.123,
            "LastUpdatedDate": 1521097200.123
        },
    ]
```

```
}


$ aws kms update-alias --alias-name alias/TestKey --target-key-id
 0987dcba-09fe-87dc-65ba-ab0987654321

$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/TestKey`]'
{
    "Aliases": [
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/TestKey",
            "AliasName": "alias/TestKey",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "CreationDate": 1521097200.123,
            "LastUpdatedDate": 1604958290.722
        },
    ]
}
```

Modifier la clé primaire dans un ensemble de clés multirégionales

Chaque ensemble de clés multi-région associées doit posséder une clé principale. Mais vous pouvez changer la clé principale. Cette action, connue sous le nom de mise à jour de la région principale, convertit la clé principale actuelle en clé de réplica et convertit l'une des clés de réplica associées en clé principale. Vous pouvez le faire si vous avez besoin de supprimer la clé principale actuelle tout en conservant les clés de réplica, ou si vous devez localiser la clé principale dans la même région que vos administrateurs de clé.

Vous pouvez sélectionner n'importe quelle clé de réplica associée comme nouvelle clé principale. La clé principale et la clé de réplica doivent être toutes les deux dans l'état de clé `Enabled` lorsque l'opération démarre.

L'état **`Updating`** clé

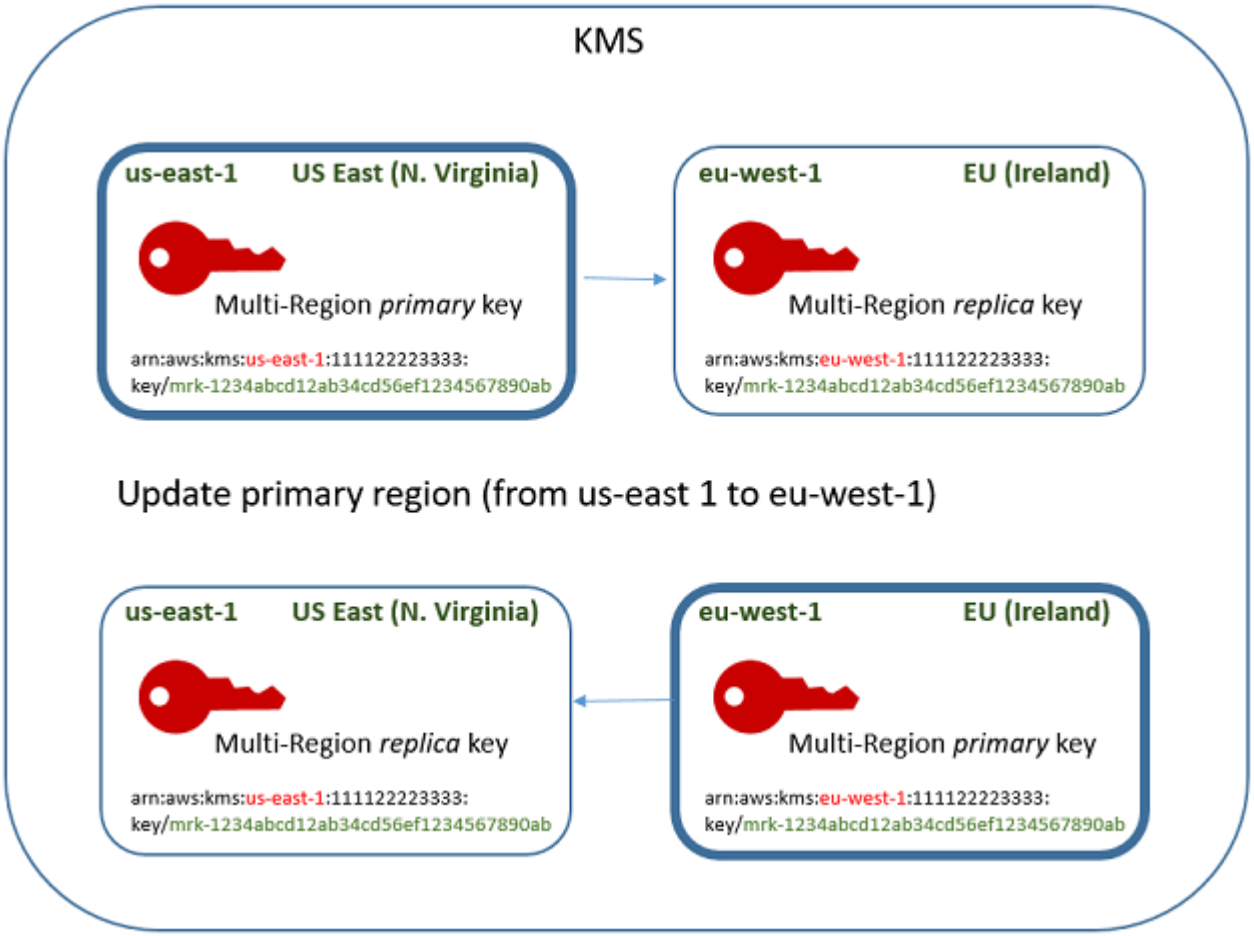
> Même une fois l'`UpdatePrimaryRegion`opération terminée, le processus de mise à jour de la région principale peut encore être en cours pendant quelques secondes. Pendant ce temps, les anciennes et les nouvelles clés principales ont un état de clé transitoire Updating (Mise à jour en cours). Lorsque l'état de la clé est `Updating`, vous pouvez utiliser les clés dans les opérations cryptographiques, mais vous ne pouvez pas répliquer la nouvelle clé principale ou effectuer certaines opérations de gestion, telles que l'activation ou la désactivation de ces clés.

Des opérations telles que celles DescribeKeysusceptibles d'afficher à la fois les anciennes et les nouvelles clés primaires sous forme de répliques. L'état de la clé `Enabled` est restauré lorsque la mise à jour est terminée.

Pour plus d'informations sur l'effet de l'état de clé `Updating`, veuillez consulter États clés des AWS KMS clés.

Comment ça marche

Supposons que vous avez une clé principale dans la région USA Est (Virginie du Nord) (us-east-1) et une clé de réplica dans la région UE (Irlande) (eu-west-1). Vous pouvez utiliser la fonction de mise à jour pour remplacer la clé principale dans la région USA Est (Virginie du Nord) (us-east-1) par une clé de réplica et transformer la clé de réplica dans la région UE (Irlande) (eu-west-1) par la clé principale.



Une fois le processus de mise à jour terminé, la clé multi-région dans la région UE (Irlande) (eu-west-1) est une clé principale multi-région et la clé dans la région USA Est (Virginie du Nord) (us-east-1) est sa clé de réplica. S'il existe d'autres clés de réplica associées, elles deviennent

des réplicas de la nouvelle clé principale. La prochaine fois qu'il AWS KMS synchronisera les propriétés partagées des clés multirégionales, il obtiendra les propriétés partagées de la nouvelle clé primaire et les copiera dans ses clés répliques, y compris l'ancienne clé primaire.

L'opération de mise à jour n'a aucun effet sur l'ARN de clé de n'importe quelle clé multi-région. Elle n'a pas non plus d'effet sur les propriétés partagées, telles que les éléments de clé, ni sur les propriétés indépendantes, telles que la politique de clé. Toutefois, vous devrez peut-être mettre à jour la politique de clé de la nouvelle clé principale. Par exemple, vous souhaiterez peut-être ajouter kms : ReplicateKey permission for trusted principals à la nouvelle clé primaire et la supprimer de la nouvelle clé de réplique.

Mettre à jour la région principale

Vous pouvez convertir une clé de réplique en clé primaire, ce qui transforme l'ancienne clé primaire en réplique. Pour mettre à jour la région principale, vous devez disposer de UpdatePrimaryRegion l'autorisation kms : dans les deux régions.

Vous pouvez mettre à jour la région principale dans la AWS KMS console ou en utilisant l'UpdatePrimaryRegionopération.

Utilisation de la AWS KMS console

Vous pouvez mettre à jour la clé primaire dans la AWS KMS console. Commencez sur la page des détails de la clé principale actuelle.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Sélectionnez l'ID de clé ou l'alias de la clé principale multi-région. La page des détails de la clé principale s'ouvre.

   Pour identifier une clé principale multi-région, utilisez l'icône de l'outil dans le coin supérieur droit pour ajouter la colonne Regionality (Régionalité) dans la table.
5. Cliquez sur l'onglet Regionality (Régionalité).
6. Dans la section Primary key (Clé principale), choisissez Change primary Region (Modifier la région principale).

7. Choisissez la région de la nouvelle clé principale. Vous ne pouvez choisir qu'une seule région dans le menu.

   Le menu Change primary Regions (Modifier les régions principales) n'inclut que les régions associées à une clé multi-région. Vous n'êtes peut-être pas autorisé à mettre à jour la région principale dans toutes les régions du menu.
8. Choisissez Change primary Region (Modifier la région principale).

Utilisation de l' AWS KMS API

Pour modifier la clé primaire dans un ensemble de clés multirégionales associées, utilisez l'UpdatePrimaryRegionopération.

Utilisez le paramètre `KeyId` pour identifier la clé principale actuelle. Utilisez le `PrimaryRegion` paramètre pour indiquer Région AWS la nouvelle clé primaire. Si la clé principale n'a pas déjà de réplica dans la nouvelle région principale, l'opération échoue.

L'exemple suivant transforme la clé principale de la clé multi-région dans la région `us-west-2` en son réplica dans la région `eu-west-1`. Le paramètre `KeyId` identifie la clé principale actuelle dans la région `us-west-2`. Le `PrimaryRegion` paramètre spécifie Région AWS la nouvelle clé primaire,`eu-west-1`.

```
$ aws kms update-primary-region \
      --key-id arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab \
      --primary-region eu-west-1
```

En cas de succès, cette opération ne renvoie aucune sortie ; seulement le code d'état HTTP. Pour voir l'effet, appelez l'DescribeKeyopération sur l'une des touches multirégions. Vous devrez peut-être attendre que l'état de la clé repasse à `Enabled`. Pendant que l'état de la clé est Updating (Mise à jour en cours), les valeurs de la clé peuvent toujours être en flux.

Par exemple, l'appel `DescribeKey` suivant obtient les détails sur la clé multi-région dans la région `eu-west-1`. La sortie indique que la clé multi-région dans la région `eu-west-1` est désormais la clé principale. La clé multi-région associée (même ID de clé) dans la région `us-west-2` est désormais une clé de réplica.

```
$ aws kms describe-key \
```

```
      --key-id arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab \

{
    "KeyMetadata": {
        "AWSAccountId": "111122223333",
        "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "CreationDate": 1609193147.831,
        "Enabled": true,
        "Description": "multi-region-key",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "Origin": "AWS_KMS",
        "KeyManager": "CUSTOMER",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "MultiRegion": true,
        "MultiRegionConfiguration": {
           "MultiRegionKeyType": "PRIMARY",
           "PrimaryKey": {
              "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
              "Region": "eu-west-1"
           },
           "ReplicaKeys": [
              {
                 "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                 "Region": "us-west-2"
              }
           ]
        }
    }
}
```

Supprimer un AWS KMS key

La suppression d'un AWS KMS key est destructrice et potentiellement dangereuse. Elle supprime les éléments de clé et toutes les métadonnées associées à la clé KMS, et elle est irréversible. Après la suppression d'une clé KMS, vous ne pouvez plus déchiffrer les données chiffrées sous cette clé, ce qui signifie que les données deviennent irrécupérables. (Les seules exceptions sont les clés de réplica multi-région et les clés asymétriques et KMS HMAC avec un élément de clé importé.) Ce risque est important pour les clés KMS asymétriques utilisées pour le chiffrement où, sans avertissement ni erreur, les utilisateurs peuvent continuer à générer des textes chiffrés avec la clé publique qui ne peuvent pas être déchiffrés une fois la clé privée supprimée. AWS KMS

Vous devez supprimer une clé KMS seulement lorsque vous êtes sûr de ne plus avoir besoin de l'utiliser. Si vous n'en êtes pas sûr, envisagez de désactiver la clé KMS au lieu de la supprimer. Vous pouvez réactiver une clé KMS désactivée et annuler la suppression planifiée d'une clé KMS, mais vous ne pouvez pas récupérer une clé KMS supprimée.

Vous ne pouvez pas planifier la suppression d'une clé gérée par le client. Vous ne pouvez pas supprimer Clés gérées par AWS ou Clés détenues par AWS.

Avant de supprimer une clé KMS, vous souhaiterez peut-être savoir combien de textes chiffrés ont été chiffrés sous cette clé KMS. AWS KMS ne stocke pas ces informations et ne stocke aucun texte chiffré. Pour obtenir ces informations, vous devez déterminer l'utilisation passée d'une clé KMS. Pour obtenir de l'aide, rendez-vous sur Déterminer l'utilisation passée d'une clé KMS.

AWS KMS ne supprime jamais vos clés KMS à moins que vous ne les programmiez explicitement pour leur suppression et que le délai d'attente obligatoire expire.

Toutefois, vous pouvez choisir de supprimer une clé KMS pour une ou plusieurs des raisons suivantes :

- Pour terminer le cycle de vie de clés KMS dont vous n'avez plus besoin
- Pour éviter les frais généraux et les coûts de gestion associés à la maintenance des clés KMS inutilisées
- Pour réduire le nombre de clés KMS qui comptent par rapport à votre quota de ressources de clés KMS

> **Note**
>
> Si vous fermez votre Compte AWS, vos clés KMS deviennent inaccessibles et ne vous sont plus facturées.

AWS KMS enregistre une entrée dans votre AWS CloudTrail journal lorsque vous planifiez la suppression de la clé KMS et lorsque la clé KMS est effectivement supprimée.

À propos de la période d'attente

Comme il est destructeur et potentiellement dangereux de supprimer une clé KMS AWS KMS , vous devez définir un délai d'attente de 7 à 30 jours. La période d'attente par défaut est de 30 jours.

Cependant, la période d'attente réelle peut être jusqu'à 24 heures plus longue celle que vous avez planifiée. Pour obtenir la date et l'heure réelles auxquelles la clé KMS sera supprimée, utilisez l'DescribeKeyopération. Ou, dans la console AWS KMS , dans la page de détails de la clé KMS, dans la section General configuration (Configuration générale), veuillez consulter la Scheduled deletion date (Date de suppression planifiée). Assurez-vous de noter le fuseau horaire.

Pendant la période d'attente, l'état de la clé KMS est Pending deletion (Suppression en attente).

- Une clé KMS qui est en attente de suppression ne peut pas être utilisée dans une opération cryptographique.
- AWS KMS ne fait pas pivoter le contenu clé des clés KMS en attente de suppression.

Une fois la période d'attente terminée, AWS KMS supprime la clé KMS, ses alias et toutes les métadonnées associées AWS KMS .

La planification de la suppression d'une clé KMS peut ne pas affecter immédiatement les clés de données chiffrées par la clé KMS. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

Utilisez la période d'attente pour vous assurer de ne pas avoir besoin de la clé KMS maintenant ni par la suite. Vous pouvez configurer une CloudWatch alarme Amazon pour vous avertir si une personne ou une application tente d'utiliser la clé KMS pendant la période d'attente. Pour récupérer la clé KMS, vous pouvez annuler la suppression de clé avant la fin de la période d'attente. Une fois

la période d'attente terminée, vous ne pouvez pas annuler la suppression de la clé et AWS KMS supprimez la clé KMS.

Considérations spéciales

Avant de planifier la suppression de vos clés, passez en revue les considérations spéciales suivantes concernant la suppression de clés KMS à usage spécifique.

Suppression de clés KMS asymétriques

Les utilisateurs autorisés peuvent supprimer des clés KMS symétriques ou asymétriques. La procédure pour planifier la suppression de ces clés KMS est la même pour les deux types de clés. Cependant, étant donné que la clé publique d'une clé KMS asymétrique peut être téléchargée et utilisée à l'extérieur AWS KMS, l'opération présente des risques supplémentaires importants, en particulier pour les clés KMS asymétriques utilisées pour le chiffrement (l'utilisation de la clé est`ENCRYPT_DECRYPT`).

- Lorsque vous planifiez la suppression d'une clé KMS, l'état de la clé KMS devient Pending deletion (Suppression en attente), et la clé ne peut pas être utilisée dans les opérations cryptographiques. Cependant, la planification de la suppression n'a aucun effet sur les clés publiques en dehors de AWS KMS. Les utilisateurs disposant de la clé publique peuvent continuer à les utiliser pour chiffrer les messages. Ils ne reçoivent aucune notification indiquant que l'état de la clé est modifié. Sauf si la suppression est annulée, le texte chiffré créé avec la clé publique ne peut pas être déchiffré.
- Les alarmes, les journaux et les autres politiques qui détectent les tentatives d'utilisation de la clé KMS en attente de suppression ne peuvent pas détecter l'utilisation de la clé publique en dehors d' AWS KMS.
- Lorsque la clé KMS est supprimée, toutes les AWS KMS actions impliquant cette clé KMS échouent. Toutefois, les utilisateurs disposant de la clé publique peuvent continuer à les utiliser pour chiffrer les messages. Ces textes chiffrés ne peuvent pas être déchiffrés.

Si vous devez supprimer une clé KMS asymétrique dont la clé d'utilisation est de`ENCRYPT_DECRYPT`, utilisez les entrées de votre CloudTrail journal pour déterminer si la clé publique a été téléchargée et partagée. Si c'est le cas, vérifiez que la clé publique n'est pas utilisée en dehors d' AWS KMS. Ensuite, pensez à désactiver la clé KMS au lieu de la supprimer.

Le risque lié à la suppression d'une clé KMS asymétrique est atténué pour les clés KMS asymétriques avec un élément de clé importé. Pour plus de détails, consultez Deleting KMS keys with imported key material.

Suppression de clés multirégionales

Pour supprimer une clé principale, vous devez planifier la suppression de toutes ses clés de réplica, puis attendre que celles-ci soient supprimées. Le délai d'attente requis pour la suppression d'une clé principale commence lorsque la dernière de ses clés de réplica est supprimée. Si vous devez supprimer une clé principale d'une région particulière sans supprimer ses clés de réplica, transformez la clé principale en clé de réplica en mettant à jour la région principale.

Vous pouvez supprimer une clé de réplica à tout moment. Cela ne dépend pas de l'état de clé d'une autre clé KMS. Si vous supprimez par erreur une clé de réplica, vous pouvez la recréer en répliquant la même clé primaire dans la même région. La nouvelle clé de réplica que vous allez créer aura les mêmes propriétés partagées que la clé de réplica d'origine.

Suppression de clés KMS avec du matériel clé importé

La suppression des éléments de clé d'une clé KMS avec des éléments de clé importés est temporaire et réversible. Pour restaurer la clé, réimportez son élément de clé.

En revanche, la suppression d'une clé KMS est irréversible. Si vous planifiez la suppression de la clé et que le délai d'attente requis expire, supprime AWS KMS définitivement et irréversiblement la clé KMS, son contenu clé et toutes les métadonnées associées à la clé KMS.

Cependant, les risques et les conséquences liés à la suppression d'une clé KMS contenant un élément de clé importé dépendent du type (« spécification de clé ») de la clé KMS.

- Clés de chiffrement symétriques – Si vous supprimez une clé KMS de chiffrement symétrique, tous les textes chiffrés restants chiffrés par cette clé sont irrécupérables. Vous ne pouvez pas créer une nouvelle clé KMS de chiffrement symétrique capable de déchiffrer les textes chiffrés d'une clé KMS de chiffrement symétrique supprimée, même si vous disposez du même élément de clé. Les métadonnées propres à chaque clé KMS sont liées par cryptographie à chaque texte chiffré symétrique. Cette fonctionnalité de sécurité garantit que seule la clé KMS qui a chiffré le texte chiffré symétrique peut le déchiffrer, mais elle vous empêche de recréer une clé KMS équivalente.
- Clés asymétriques et HMAC : si vous disposez du contenu de la clé d'origine, vous pouvez créer une nouvelle clé KMS avec les mêmes propriétés cryptographiques qu'une clé asymétrique ou HMAC KMS supprimée. AWS KMS génère des textes chiffrés et des signatures RSA standard, des signatures ECC et des balises HMAC, qui n'incluent aucune fonctionnalité de sécurité unique. Vous pouvez également utiliser une clé HMAC ou la clé privée d'une paire de clés asymétriques à l'extérieur de AWS.

Une nouvelle clé KMS que vous créez avec le même élément de clé asymétrique ou HMAC aura un identifiant de clé différent. Vous devrez créer une nouvelle stratégie de clé, recréer tous les alias et mettre à jour les politiques et autorisations IAM existantes pour faire référence à la nouvelle clé.

Supprimer des clés KMS d'un magasin de AWS CloudHSM clés

Lorsque vous planifiez la suppression d'une clé KMS d'un magasin de AWS CloudHSM clés, son état de clé passe à En attente de suppression. La clé KMS reste à l'état Pending deletion (En attente de suppression) tout au long de la période d'attente, même si la clé KMS n'est pas disponible parce que vous avez déconnecté la clé personnalisée. Cela vous permet d'annuler la suppression de la clé KMS à tout moment au cours de la période d'attente.

Lorsque le délai d'attente expire, AWS KMS supprime la clé KMS de AWS KMS. AWS KMS Fait ensuite de son mieux pour supprimer le matériel clé du AWS CloudHSM cluster associé. Si AWS KMS ne peut pas supprimer les clés, comme lorsque, par exemple, le magasin de clés est déconnecté de AWS KMS, il se peut que vous ayez besoin de supprimer manuellement les clé orphelines du cluster.

AWS KMS ne supprime pas le matériel clé des sauvegardes du cluster. Même si vous supprimez la clé KMS AWS KMS et son contenu clé de votre AWS CloudHSM cluster, les clusters créés à partir de sauvegardes peuvent contenir le contenu clé supprimé. Pour supprimer définitivement le contenu clé, utilisez l'DescribeKeyopération pour identifier la date de création de la clé KMS. Ensuite, supprimez toutes les sauvegardes de cluster qui peuvent contenir la clé.

Lorsque vous planifiez la suppression d'une clé KMS d'un magasin de AWS CloudHSM clés, la clé KMS devient immédiatement inutilisable (sous réserve de cohérence éventuelle). Toutefois, les ressources chiffrées à l'aide de clés de données protégées par la clé KMS ne sont pas affectées tant que la clé KMS n'est pas réutilisée, par exemple pour déchiffrer la clé de données. Ce problème concerne la Services AWS plupart d'entre eux qui utilisent des clés de données pour protéger vos ressources. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

Supprimer des clés KMS d'un magasin de clés externe

La suppression d'une clé KMS d'un magasin de clés externe n'a aucun effet sur la clé externe qui lui a servi d'élément de clé.

Lorsque vous planifiez la suppression d'une clé KMS d'un magasin de clés externe, son état de clé passe à Pending deletion (Suppression en attente). La clé KMS reste à l'état Pending

deletion (Suppression en attente) tout au long de la période d'attente, même si la clé KMS n'est pas disponible parce que vous avez déconnecté le magasin de clés externe. Cela vous permet d'annuler la suppression de la clé KMS à tout moment au cours de la période d'attente. Lorsque le délai d'attente expire, AWS KMS supprime la clé KMS de AWS KMS.

Lorsque vous planifiez la suppression d'une clé KMS d'un magasin de clés externe, la clé KMS devient immédiatement inutilisable (sous réserve d'une éventuelle cohérence). Toutefois, les ressources chiffrées à l'aide de clés de données protégées par la clé KMS ne sont pas affectées tant que la clé KMS n'est pas réutilisée, par exemple pour déchiffrer la clé de données. Ce problème affecte les Services AWS, dont beaucoup utilisent des clés de données pour protéger vos ressources. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

Contrôler l'accès à la suppression des clés

Si vous utilisez des politiques IAM pour accorder AWS KMS des autorisations, les identités IAM disposant d'un accès AWS administrateur (`"Action": "*"`) ou d'un accès AWS KMS complet (`"Action": "kms:*"`) sont déjà autorisées à planifier et à annuler la suppression des clés KMS. Pour permettre aux administrateurs clés de planifier et d'annuler la suppression des clés dans la politique des clés, utilisez la AWS KMS console ou l' AWS KMS API.

En général, seuls les administrateurs de clés sont autorisés à planifier ou à annuler la suppression des clés. Vous pouvez toutefois accorder ces autorisations à d'autres identités IAM en ajoutant les autorisations `kms:ScheduleKeyDeletion` et `kms:CancelKeyDeletion` à la politique de clé ou à une politique IAM. Vous pouvez également utiliser la clé de `kms:ScheduleKeyDeletionPendingWindowInDays`condition pour restreindre davantage les valeurs que les principaux peuvent spécifier dans le `PendingWindowInDays` paramètre d'une ScheduleKeyDeletiondemande.

Permettre aux administrateurs clés de planifier et d'annuler la suppression des clés

Utilisation de la console AWS KMS

Autoriser les administrateurs de clés à planifier et annuler une suppression de clé.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.

2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez l'alias ou l'ID de clé de la clé KMS dont vous voulez modifier les autorisations.
5. Choisissez l'onglet Key policy (Politique de clé).
6. L'étape suivante diffère en ce qui concerne l'affichage par défaut et l'affichage des politiques de votre politique de clé. La vue par défaut n'est disponible que si vous utilisez la politique de clé de console par défaut. Dans le cas contraire, seul l'affichage des politiques est disponible.

   Lorsque la vue par défaut est disponible, un bouton Switch to policy view (Passer à la vue de politique) ou Switch to default view (Passer à la vue par défaut) apparaît dans l'onglet Key policy (Politique de clé).

   - Dans la vue par défaut :
     - Sous Key deletion (Suppression de clé), sélectionnez Allow key administrators to delete this key (Autoriser les administrateurs de clé à supprimer cette clé).
   - Dans la vue de politique :
     a. Choisissez Edit (Modifier).
     b. Dans l'instruction de politique destinée aux administrateurs de clés, ajoutez les autorisations `kms:ScheduleKeyDeletion` et `kms:CancelKeyDeletion` à l'élément `Action`.

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSKeyAdmin"},
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:ScheduleKeyDeletion",
```

```
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

c. Sélectionnez Enregistrer les modifications.

Utilisation de l' AWS KMS API

Vous pouvez utiliser le AWS Command Line Interface pour ajouter des autorisations de planification et d'annulation de la suppression de clés.

Pour ajouter une autorisation pour planifier et annuler une suppression de clé

1. Utilisez la commande `aws kms get-key-policy` pour récupérer la politique de clé existante, puis enregistrez le document de politique dans un fichier.
2. Ouvrez le document de stratégie dans votre éditeur de texte préféré. Dans l'instruction de politique destinée aux administrateurs de clés, ajoutez les autorisations `kms:ScheduleKeyDeletion` et `kms:CancelKeyDeletion`. L'exemple suivant montre une déclaration de politique avec les deux autorisations suivantes :

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSKeyAdmin"},
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

3. Utilisez la commande `aws kms put-key-policy` pour appliquer la politique de clé à la clé KMS.

Planifier la suppression des clés

Les procédures suivantes décrivent comment planifier la suppression des clés et annuler la suppression des clés AWS KMS keys (clés KMS) à AWS KMS l'aide de l'API AWS Management Console et de l' AWS KMS API.

> **Warning**
>
> La suppression d'une clé KMS est destructrice et potentiellement dangereuse. Vous devez y avoir recours seulement lorsque vous êtes sûr de ne plus avoir besoin d'utiliser la clé KMS maintenant ni par la suite. Si vous n'en êtes pas sûr, vous devriez désactiver la clé KMS au lieu de la supprimer.

Avant de pouvoir supprimer une clé KMS, vous devez avoir la permission de le faire. Pour plus d'informations sur l'octroi de ces autorisations aux administrateurs de clés, veuillez consulter la rubrique Contrôler l'accès à la suppression des clés. Vous pouvez également utiliser la clé de condition `kms:ScheduleKeyDeletionPendingWindowInDays` pour limiter davantage le délai d'attente, par exemple en imposant un délai d'attente minimum.

AWS KMS enregistre une entrée dans votre AWS CloudTrail journal lorsque vous planifiez la suppression de la clé KMS et lorsque la clé KMS est effectivement supprimée.

Utilisation de la console AWS KMS

Dans le AWS Management Console, vous pouvez planifier et annuler la suppression de plusieurs clés KMS à la fois.

Pour planifier une suppression de clé

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.

Vous ne pouvez pas planifier la suppression deClés gérées par AWS ou de Clés détenues par AWS.

4. Cochez la case en regard de la clé KMS que vous souhaitez supprimer.
5. Choisissez Actions de clé, Planifier une suppression de clé.
6. Lisez et tenez compte de l'avertissement et des informations sur l'annulation de la suppression pendant la période d'attente. Si vous décidez d'annuler la suppression, en bas de la page, sélectionnez Cancel (Annuler).
7. Pour Période d'attente (en jours), tapez un nombre de jours compris entre 7 et 30.
8. Vérifiez les clés KMS que vous supprimez.
9. Cochez la case à côté de Confirmer que vous souhaitez planifier la suppression de cette clé dans ***<number of days>*** quelques jours. .
10. Choisissez Schedule deletion (Planifier la suppression).

L'état de la clé KMS passe à Pending deletion (En attente de suppression).

Utilisation de l'API AWS KMS

Utilisez la commande `aws kms schedule-key-deletion` pour planifier une suppression de clé gérée par le client, comme illustré dans l'exemple suivant.

Vous ne pouvez pas planifier la suppression d'un Clé gérée par AWS ou Clé détenue par AWS.

```
$ aws kms schedule-key-deletion --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --
pending-window-in-days 10
```

Lorsqu'il est utilisé avec succès, le résultat AWS CLI renvoyé est similaire à celui illustré dans l'exemple suivant :

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "DeletionDate": 1598304792.0,
    "KeyState": "PendingDeletion",
    "PendingWindowInDays": 10
}
```

Annuler la suppression de la clé

Après avoir planifié la suppression d'une clé KMS, vous pouvez annuler la suppression de la clé tant qu'elle est toujours en attente de suppression. Vous pouvez annuler la suppression de la clé dans la AWS KMS console ou en utilisant cette CancelKeyDeletionopération. Après avoir annulé la suppression en attente d'une clé KMS, l'état de la clé KMS est`Disabled`. Pour plus d'informations sur l'activation de la clé KMS, consultezActiver et désactiver les touches.

Utilisation de la AWS KMS console

Pour annuler une suppression de clé

1. Ouvrez la AWS KMS console à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Cochez la case en regard de la clé KMS que vous souhaitez récupérer.
5. Choisissez Actions de clé, Annuler la suppression de clé.

L'état de la clé KMS passe de Pending deletion (En attente de suppression) à Disabled (Désactivé). Pour utiliser la clé KMS, vous devez l'activer.

Utilisation de l' AWS KMS API

Utilisez la `aws kms cancel-key-deletion`commande pour annuler la suppression de la touche AWS CLI , comme indiqué dans l'exemple suivant.

```
$ aws kms cancel-key-deletion --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Lorsqu'il est utilisé avec succès, le résultat AWS CLI renvoyé est similaire à celui illustré dans l'exemple suivant :

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

L'état de la clé KMS passe de Pending Deletion (En attente de suppression) à Disabled (Désactivé). Pour utiliser la clé KMS, vous devez l'activer.

Créez une alarme qui détecte l'utilisation d'une clé KMS en attente de suppression

Vous pouvez combiner les fonctionnalités d' AWS CloudTrail Amazon CloudWatch Logs et d'Amazon Simple Notification Service (Amazon SNS) pour créer une alarme CloudWatch Amazon qui vous avertit lorsqu'un utilisateur de votre compte essaie d'utiliser une clé KMS en attente de suppression. Si vous recevez cette notification, vous pouvez annuler la suppression de la clé KMS et reconsidérer votre décision de la supprimer.

Les procédures suivantes créent une alarme qui vous avertit chaque fois que le message d'erreur *`Key ARN`* `is pending deletion` « » est écrit dans vos fichiers CloudTrail journaux. Ce message d'erreur indique qu'une personne ou une application tente d'utiliser la clé KMS dans une opération de chiffrement. Étant donné que la notification est liée au message d'erreur, elle n'est pas déclenchée lorsque vous utilisez des opérations d'API autorisées sur les clés KMS en attente de suppression, telles que `ListKeys`, `CancelKeyDeletion` et `PutKeyPolicy`. Pour consulter la liste des opérations d' AWS KMS API qui renvoient ce message d'erreur, consultezÉtats clés des AWS KMS clés.

L'e-mail de notification que vous recevez ne répertorie pas la clé KMS ou les opérations de chiffrement. Vous pouvez trouver ces informations dans votre journal CloudTrail. Au lieu de cela, l'e-mail indique que l'état de l'alarme est passé de OK à Alarme. Pour plus d'informations sur les CloudWatch alarmes et les changements d'état, consultez la section Utilisation des CloudWatch alarmes Amazon dans le guide de CloudWatch l'utilisateur Amazon.

> **Warning**
>
> Cette CloudWatch alarme Amazon ne peut pas détecter l'utilisation de la clé publique d'une clé KMS asymétrique en dehors de AWS KMS. Pour plus de détails sur les risques particuliers liés à la suppression de clés KMS asymétriques utilisées pour le chiffrement de la clé publique, en particulier la création de textes chiffrés qui ne peuvent pas être déchiffrés, veuillez consulter Deleting asymmetric KMS keys.

Dans cette procédure, vous créez un filtre métrique de groupe de CloudWatch journaux qui recherche les instances de l'exception de suppression en attente. Vous créez ensuite une CloudWatch alarme

en fonction de la métrique du groupe de logs. Pour plus d'informations sur les filtres métriques des groupes de journaux, consultez la section Création de métriques à partir d'événements de journal à l'aide de filtres dans le guide de l'utilisateur Amazon CloudWatch Logs.

1. Créez un filtre CloudWatch métrique qui analyse les CloudTrail journaux.

   Suivez les instructions de la section Créer un filtre métrique pour un groupe de journaux à l'aide des valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Modèle de filtre	`{ $.eventSource = kms* && $.errorMessage = "* is pending deletion."}`
Valeur de la métrique	1

2. Créez une CloudWatch alarme en fonction du filtre métrique que vous avez créé à l'étape 1.

   Suivez les instructions de la section Création d'une CloudWatch alarme basée sur un filtre métrique de groupes de logs en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Filtre de métrique	Le nom du filtre de métrique que vous avez créé à l'étape 1.
Type de seuil	Statique
Conditions	Chaque fois que *`metric-name`* c'est supérieur/égal à 1
Points de données indiquant une alarme	1 sur 1

Champ	Valeur
Traitement de données manquantes	Traiter les données manquantes comme correctes (seuil non dépassé)

Une fois cette procédure terminée, vous recevrez une notification chaque fois que votre nouvelle CloudWatch alarme entre dans l'`ALARM`état. Si vous recevez une notification pour cette alarme, cela peut signifier qu'une clé KMS dont la suppression est planifiée est toujours nécessaire pour chiffrer ou déchiffrer les données. Dans ce cas, annulez la suppression de la clé KMS et reconsidérez votre décision de la supprimer.

Déterminer l'utilisation passée d'une clé KMS

Avant de supprimer une clé KMS, vous souhaiterez peut-être savoir combien de textes chiffrés ont été chiffrés sous cette clé. AWS KMS ne stocke pas ces informations et ne stocke aucun des textes chiffrés. Savoir comment une clé KMS a été utilisée dans le passé peut vous aider à décider si vous en aurez besoin ou non à l'avenir. Cette rubrique propose plusieurs politiques qui peuvent vous aider à déterminer l'utilisation passée d'une clé KMS.

> **Warning**
>
> Ces stratégies visant à déterminer l'utilisation passée et réelle ne sont efficaces que pour AWS les utilisateurs et les AWS KMS opérations. Elles ne peuvent pas détecter l'utilisation de la clé publique d'une clé KMS asymétrique en dehors d' AWS KMS. Pour plus de détails sur les risques particuliers liés à la suppression de clés KMS asymétriques utilisées pour le chiffrement de la clé publique, en particulier la création de textes chiffrés qui ne peuvent pas être déchiffrés, veuillez consulter Deleting asymmetric KMS keys.

Rubriques

- Examinez les autorisations clés KMS pour déterminer l'étendue de l'utilisation potentielle
- Examiner AWS CloudTrail les journaux pour déterminer l'utilisation réelle

Examinez les autorisations clés KMS pour déterminer l'étendue de l'utilisation potentielle

Déterminer qui a actuellement accès à une clé KMS peut vous aider à déterminer l'ampleur de l'utilisation passée de cette clé KMS et si elle est encore requise. Pour découvrir comment déterminer qui a actuellement accès à une clé KMS, consultez la rubrique Déterminer l'accès à AWS KMS keys.

Examiner AWS CloudTrail les journaux pour déterminer l'utilisation réelle

Vous pouvez éventuellement utiliser un historique d'utilisation de clé KMS pour déterminer si vous disposez de textes chiffrés sous une clé KMS particulière.

Toutes les activités de l' AWS KMS API sont enregistrées dans des fichiers AWS CloudTrail journaux. Si vous avez créé un suivi CloudTrail dans la région où se trouve votre clé KMS, vous pouvez examiner vos fichiers CloudTrail journaux pour consulter l'historique de toutes les activités d' AWS KMS API relatives à une clé KMS en particulier. Si vous n'avez pas de parcours, vous pouvez toujours consulter les événements récents dans l'historique de vos CloudTrail événements. Pour plus de détails sur la façon dont AWS KMS les utilisations sont CloudTrail utilisées, voirJournalisation des appels d' AWS KMS API avec AWS CloudTrail.

Les exemples suivants montrent les entrées de CloudTrail journal générées lorsqu'une clé KMS est utilisée pour protéger un objet stocké dans Amazon Simple Storage Service (Amazon S3). Dans cet exemple, l'objet est chargé vers Simple Storage Service (Amazon S3) au moyen de la Protection des données à l'aide du chiffrement côté serveur avec des clés KMS (SSE-KMS). Lorsque vous chargez un objet sur Amazon S3 avec SSE-KMS, vous spécifiez la clé KMS à utiliser pour protéger l'objet. Amazon S3 utilise AWS KMS GenerateDataKeyopération pour demander une clé de données unique pour l'objet, et cet événement de demande est enregistré CloudTrail avec une entrée similaire à la suivante :

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROACKCEVSQ6C2EXAMPLE:example-user",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admins/example-user",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
```

```
        "creationDate": "2015-09-10T23:12:48Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/Admins",
        "accountId": "111122223333",
        "userName": "Admins"
      }
    },
    "invokedBy": "internal.amazonaws.com"
  },
  "eventTime": "2015-09-10T23:58:18Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "internal.amazonaws.com",
  "userAgent": "internal.amazonaws.com",
  "requestParameters": {
    "encryptionContext": {"aws:s3:arn": "arn:aws:s3:::example_bucket/example_object"},
    "keySpec": "AES_256",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "requestID": "cea04450-5817-11e5-85aa-97ce46071236",
  "eventID": "80721262-21a5-49b9-8b63-28740e7ce9c9",
  "readOnly": true,
  "resources": [{
    "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Lorsque vous téléchargerez ultérieurement cet objet depuis Amazon S3, Amazon S3 envoie une `Decrypt` demande AWS KMS pour déchiffrer la clé de données de l'objet à l'aide de la clé KMS spécifiée. Dans ce cas, vos fichiers CloudTrail journaux incluent une entrée similaire à la suivante :

```
{
  "eventVersion": "1.02",
```

```
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROACKCEVSQ6C2EXAMPLE:example-user",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admins/example-user",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2015-09-10T23:12:48Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/Admins",
        "accountId": "111122223333",
        "userName": "Admins"
      }
    },
    "invokedBy": "internal.amazonaws.com"
  },
  "eventTime": "2015-09-10T23:58:39Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "internal.amazonaws.com",
  "userAgent": "internal.amazonaws.com",
  "requestParameters": {
    "encryptionContext": {"aws:s3:arn": "arn:aws:s3:::example_bucket/example_object"}},
  "responseElements": null,
  "requestID": "db750745-5817-11e5-93a6-5b87e27d91a0",
  "eventID": "ae551b19-8a09-4cfc-a249-205ddba330e3",
  "readOnly": true,
  "resources": [{
    "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Toutes les activités de l' AWS KMS API sont enregistrées par CloudTrail. En évaluant ces entrées de journal, vous pouvez éventuellement déterminer l'utilisation passée d'une clé KMS particulière et cela peut vous aider à déterminer si vous souhaitez la supprimer ou non.

Pour voir d'autres exemples illustrant la façon dont l'activité des AWS KMS API apparaît dans vos fichiers CloudTrail journaux, rendez-vous surJournalisation des appels d' AWS KMS API avec AWS CloudTrail. Pour plus d'informations à ce sujet, CloudTrail consultez le guide de AWS CloudTrail l'utilisateur.

Supprimer le matériel clé importé

Vous pouvez supprimer des éléments de clé importés d'une clé KMS à tout moment. De même, lorsque le matériel clé importé avec une date d'expiration expire, le AWS KMS matériel clé est supprimé. Dans les deux cas, lorsque le contenu clé est supprimé, l'état de la clé KMS passe à En attente d'importation, et la clé KMS ne peut être utilisée dans aucune opération cryptographique.

Les clés de chiffrement symétriques à région unique peuvent être associées à plusieurs éléments clés et la suppression ou l'expiration de tout élément clé dans un état autre que le fait passer l'état de `PENDING_ROTATION` la clé en attente d'importation. Pour ces clés, KMS attribue un identifiant unique à chaque élément clé. Vous pouvez utiliser l'ListKeyRotationsAPI pour consulter ces identificateurs de matériaux clés. Vous pouvez supprimer un élément clé spécifique en spécifiant son identifiant à l'aide du `key-material-id` paramètre de l'DeleteImportedKeyMaterialAPI.

> **Warning**
>
> Le `key-material-id` paramètre est facultatif et si vous ne le spécifiez pas, il AWS KMS supprimera le matériel clé actuel.

Outre la désactivation de la clé KMS et le retrait des autorisations, la suppression des éléments de clé peut être utilisée comme stratégie pour arrêter rapidement, mais temporairement, l'utilisation de la clé KMS. En revanche, la planification de la suppression d'une clé KMS avec un élément de clé importé arrête également rapidement l'utilisation de la clé KMS. Toutefois, si la suppression n'est pas annulée pendant la période d'attente, la clé KMS, les éléments clés associés et toutes les métadonnées clés sont définitivement supprimés. Pour en savoir plus, consultez Deleting KMS keys with imported key material.

Pour supprimer des éléments clés, vous pouvez utiliser la AWS KMS console ou l'opération DeleteImportedKeyMaterialAPI. AWS KMS enregistre une entrée dans votre AWS CloudTrail journal

lorsque vous supprimez du matériel clé importé et lorsque vous AWS KMS supprimez du matériel clé expiré.

Comment la suppression de documents clés affecte les AWS services

Lorsque vous supprimez un élément clé, la clé KMS devient immédiatement inutilisable (sous réserve de cohérence éventuelle). Toutefois, les ressources chiffrées à l'aide de clés de données protégées par la clé KMS ne sont pas affectées tant que la clé KMS n'est pas réutilisée, par exemple pour déchiffrer la clé de données. Ce problème affecte les Services AWS, dont beaucoup utilisent des clés de données pour protéger vos ressources. Pour en savoir plus, consultez Comment les clés KMS inutilisables affectent les clés de données.

Utilisation de la AWS KMS console

Vous pouvez utiliser la AWS KMS console pour supprimer des éléments clés.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Effectuez l’une des actions suivantes :
   - Cochez la case correspondant à une clé KMS avec les éléments de clé importés. Choisissez Key actions, Delete key material. Pour les clés de chiffrement symétriques associées à plusieurs éléments clés, cela supprimera le contenu clé actuel.
   - Pour les clés KMS de chiffrement symétrique à région unique avec du matériel clé importé, choisissez l'alias ou l'ID de clé d'une clé KMS. Choisissez l'onglet Matériau clé et rotations. Le tableau des matériaux clés répertorie tous les matériaux clés associés à la clé. Choisissez Supprimer le matériel clé dans le menu Actions sur la ligne correspondant au matériau clé que vous souhaitez supprimer.
5. Confirmez que vous souhaitez supprimer les éléments de clé, puis choisissez Delete key material. Le statut de la clé KMS, qui correspond à son état de clé, passe à Pending import (En attente d'importation). Si le contenu clé supprimé était en bon `PENDING_ROTATION` état, le statut de la clé KMS n'est pas modifié.

Utilisation de l' AWS KMS API

Pour utiliser l'AWS KMS API afin de supprimer du contenu clé, envoyez une DeleteImportedKeyMaterialdemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI.

Remplacez *1234abcd-12ab-34cd-56ef-1234567890ab* par l'ID de clé de la clé KMS dont vous souhaitez supprimer les éléments de clé. Vous pouvez utiliser l'ID de clé ou le nom ARN de la clé KMS, mais vous ne pouvez pas utiliser un alias pour cette opération. La commande suivante supprime le matériau clé actuel qui peut être le seul élément clé associé à la clé.

```
$ aws kms delete-imported-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Pour supprimer un matériau clé spécifique, spécifiez le matériau clé identifié à l'aide du `key-material-id` paramètre. *123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0*Remplacez-le par l'identifiant du matériel clé que vous souhaitez supprimer.

```
$ aws kms delete-imported-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-material-id 123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0
```

Générer des clés de données

Les clés de données sont des clés symétriques que vous pouvez utiliser pour chiffrer des données, y compris de grandes quantités de données et d'autres clés de chiffrement des données. Contrairement aux clés KMS symétriques, qui ne peuvent pas être téléchargées, les clés de données vous sont renvoyées pour une utilisation en dehors de AWS KMS.

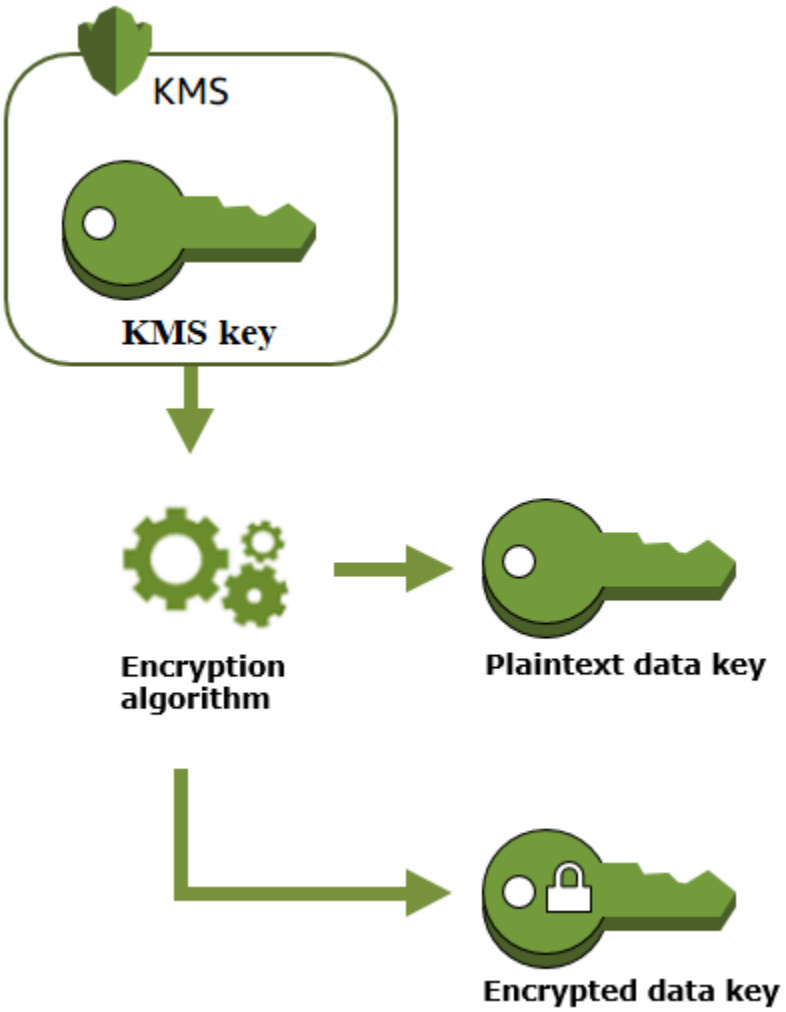
Lorsqu'il AWS KMS génère des clés de données, il renvoie une clé de données en texte brut pour une utilisation immédiate (facultatif) et une copie cryptée de la clé de données que vous pouvez stocker en toute sécurité avec les données. Lorsque vous êtes prêt à déchiffrer les données, vous demandez AWS KMS d'abord de déchiffrer la clé de données cryptée.

AWS KMS génère, chiffre et déchiffre les clés de données. Toutefois, il AWS KMS ne stocke, ne gère ni ne suit vos clés de données, et n'effectue pas d'opérations cryptographiques avec des clés de données. Vous devez utiliser et gérer les clés de données en dehors de AWS KMS. Pour obtenir de l'aide sur l'utilisation des clés de données en toute sécurité, consultez AWS Encryption SDK.

Rubriques

Création d'une clé de données

Pour créer une clé de données, appelez l'GenerateDataKeyopération. AWS KMS génère la clé de données. Il chiffre ensuite une copie de la clé de données sous une clé KMS de chiffrement symétrique que vous spécifiez. Cette opération renvoie une copie en texte clair de la clé de données et la copie de la clé de données chiffrée sous la clé KMS. L'image suivante illustre cette opération.



AWS KMS prend également en charge l'GenerateDataKeyWithoutPlaintextopération, qui renvoie uniquement une clé de données cryptée. Lorsque vous devez utiliser la clé de données, demandez AWS KMS à la déchiffrer.

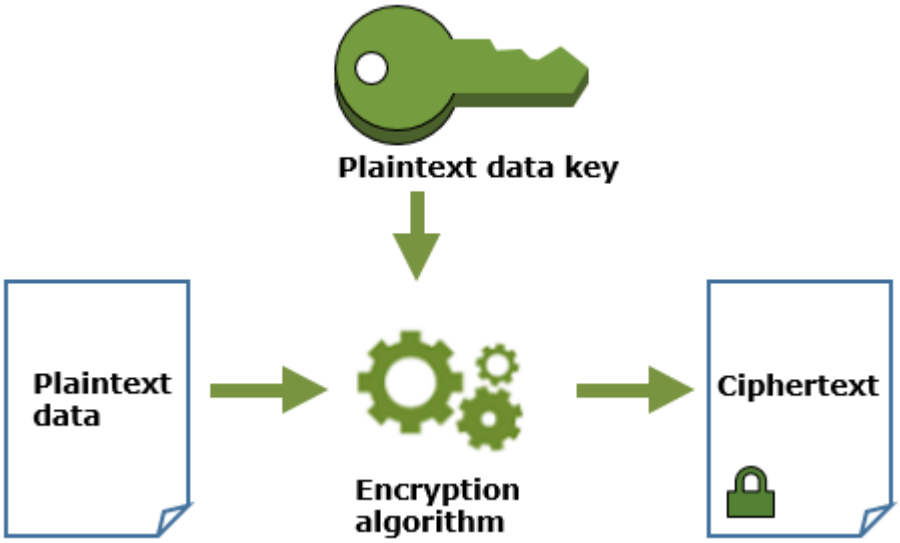
Comment fonctionnent les opérations cryptographiques avec des clés de données

Les rubriques suivantes expliquent le fonctionnement des clés de données générées par une GenerateDataKeyWithoutPlaintextopération GenerateDataKeyOR.

Chiffrement de données avec une clé de données

AWS KMS Impossible d'utiliser une clé de données pour chiffrer des données. Mais vous pouvez utiliser la clé de données en dehors de AWS KMS, par exemple en utilisant OpenSSL ou une bibliothèque cryptographique telle que. AWS Encryption SDK

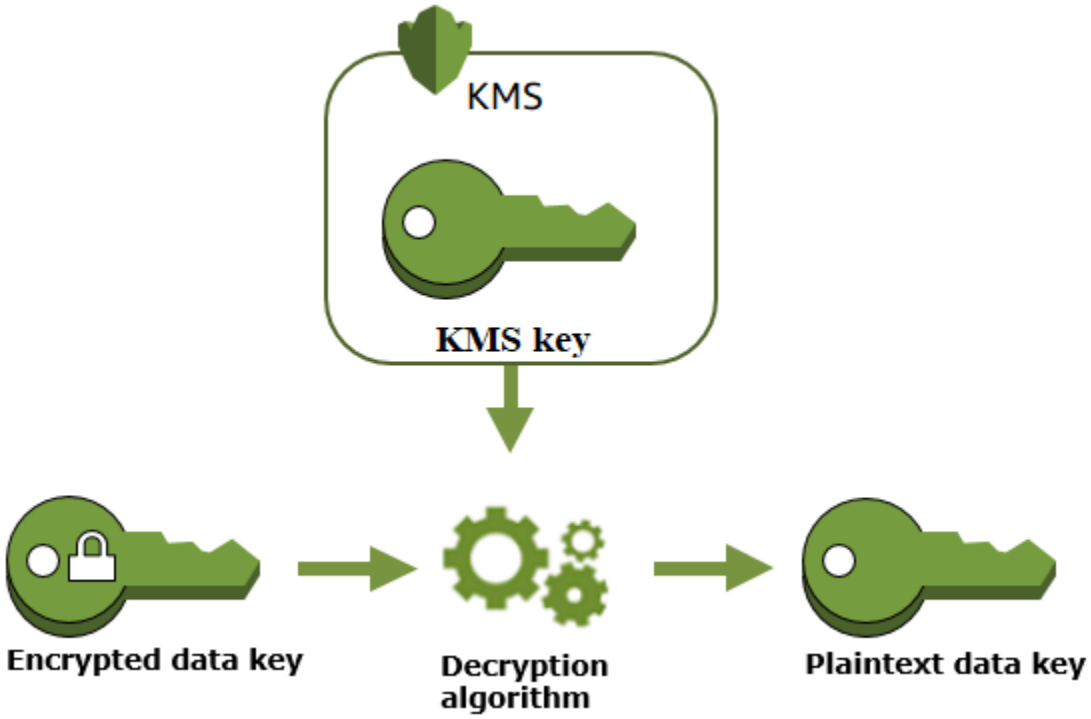
Après avoir utilisé la clé de données en texte brut pour chiffrer les données, supprimez-la de la mémoire dès que possible. Vous pouvez stocker en toute sécurité la clé de données chiffrée avec les données chiffrées pour qu'elle soit disponible pour déchiffrer les données.



Déchiffrement des données avec une clé de données

Pour déchiffrer vos données, transmettez la clé de données cryptée à l'opération de déchiffrement. AWS KMS utilise votre clé KMS pour déchiffrer la clé de données, puis renvoie la clé de données en texte brut. Utilisez la clé de données en texte brut pour déchiffrer vos données, puis supprimez la clé de données en texte brut de la mémoire dès que possible.

Le schéma suivant montre comment utiliser l'opération `Decrypt` pour déchiffrer une clé de données chiffrée.



Comment les clés KMS inutilisables affectent les clés de données

Lorsqu'une clé KMS devient inutilisable, l'effet est presque immédiat (sous réserve d'une éventuelle cohérence). L'état de clé de la clé KMS change pour refléter son nouvel état, et toutes les requêtes d'utilisation de la clé KMS dans des opérations cryptographiques échouent.

Cependant, l'effet sur les clés de données chiffrées par la clé KMS, et sur les données chiffrées par la clé de données, est retardé jusqu'à ce que la clé KMS soit utilisée à nouveau, par exemple pour déchiffrer la clé de données.

Les clés KMS peuvent devenir inutilisables pour diverses raisons, notamment les actions suivantes que vous pourriez effectuer.

- Désactiver la clé KMS
- Planifier la suppression de la clé KMS
- Supprimer les éléments de clé d'une clé KMS avec des éléments de clé importés, ou laisser les éléments de clé importés expirer Si une clé KMS avec `EXTERNAL` origine est associée à plusieurs éléments clés, la suppression ou l'expiration de tout élément clé rendra la clé inutilisable.
- Déconnecter le magasin de AWS CloudHSM clés qui héberge la clé KMS ou supprimer la clé du AWS CloudHSM cluster qui sert de matériau clé pour la clé KMS.

- Déconnecter le magasin de clés externe qui héberge la clé KMS, ou toute autre action qui interfère avec les requêtes de chiffrement et de déchiffrement adressées au proxy de magasin de clés externe, y compris la suppression de la clé externe de son gestionnaire de clés externe

Cet effet est particulièrement important pour les nombreuses personnes Services AWS qui utilisent des clés de données pour protéger les ressources gérées par le service. L'exemple suivant utilise Amazon Elastic Block Store (Amazon EBS) et Amazon Elastic Compute Cloud (EC2Amazon). Les utilisateurs Services AWS utilisent les clés de données de différentes manières. Pour plus de détails, veuillez consulter la section Protection des données du chapitre Sécurité pour l' Service AWS.

Par exemple, envisagez le scénario suivant :

1. Vous créez un volume EBS chiffré et spécifiez une clé KMS pour le protéger. Amazon EBS demande à AWS KMS d'utiliser votre clé KMS pour générer une clé de données chiffrée pour le volume. Amazon EBS stocke la clé de données chiffrée avec les métadonnées du volume.
2. Lorsque vous attachez le volume EBS à une EC2 instance, Amazon EC2 utilise votre clé KMS pour déchiffrer la clé de données chiffrée du volume EBS. Amazon EC2 utilise la clé de données du matériel Nitro, qui est chargé de chiffrer tous les disques sur I/O le volume EBS. La clé de données est conservée dans le matériel Nitro tant que le volume EBS est attaché à l' EC2 instance.
3. Vous effectuez une action qui rend la clé KMS inutilisable. Cela n'a aucun effet immédiat sur l' EC2 instance ou le volume EBS. Amazon EC2 utilise la clé de données, et non la clé KMS, pour chiffrer l'ensemble du disque I/O lorsque le volume est attaché à l'instance.
4. Toutefois, lorsque le volume EBS chiffré est détaché de l' EC2 instance, Amazon EBS supprime la clé de données du matériel Nitro. La prochaine fois que le volume EBS chiffré est attaché à une EC2 instance, la pièce jointe échoue, car Amazon EBS ne peut pas utiliser la clé KMS pour déchiffrer la clé de données chiffrée du volume. Pour utiliser le volume EBS à nouveau, vous devez rendre la clé KMS à nouveau utilisable.

Générer des paires de clés de données

Une clé KMS asymétrique représente une paire de clés de données. Les paires de clés de données sont des clés de données asymétriques composées d'une clé publique et d'une clé privée mathématiquement liées entre elles. Ils sont conçus pour être utilisés dans le chiffrement et le déchiffrement côté client, la signature et la vérification en dehors de AWS KMS ou pour établir un secret partagé entre deux pairs.

Contrairement aux paires de clés de données générées par des outils tels qu'OpenSSL AWS KMS , la clé privée de chaque paire de clés de données est protégée par une clé AWS KMS KMS de chiffrement symétrique que vous spécifiez. Toutefois, il AWS KMS ne stocke, ne gère ni ne suit vos paires de clés de données, et n'effectue pas d'opérations cryptographiques avec des paires de clés de données. Vous devez utiliser et gérer les paires de clés de données en dehors d' AWS KMS.

Rubriques

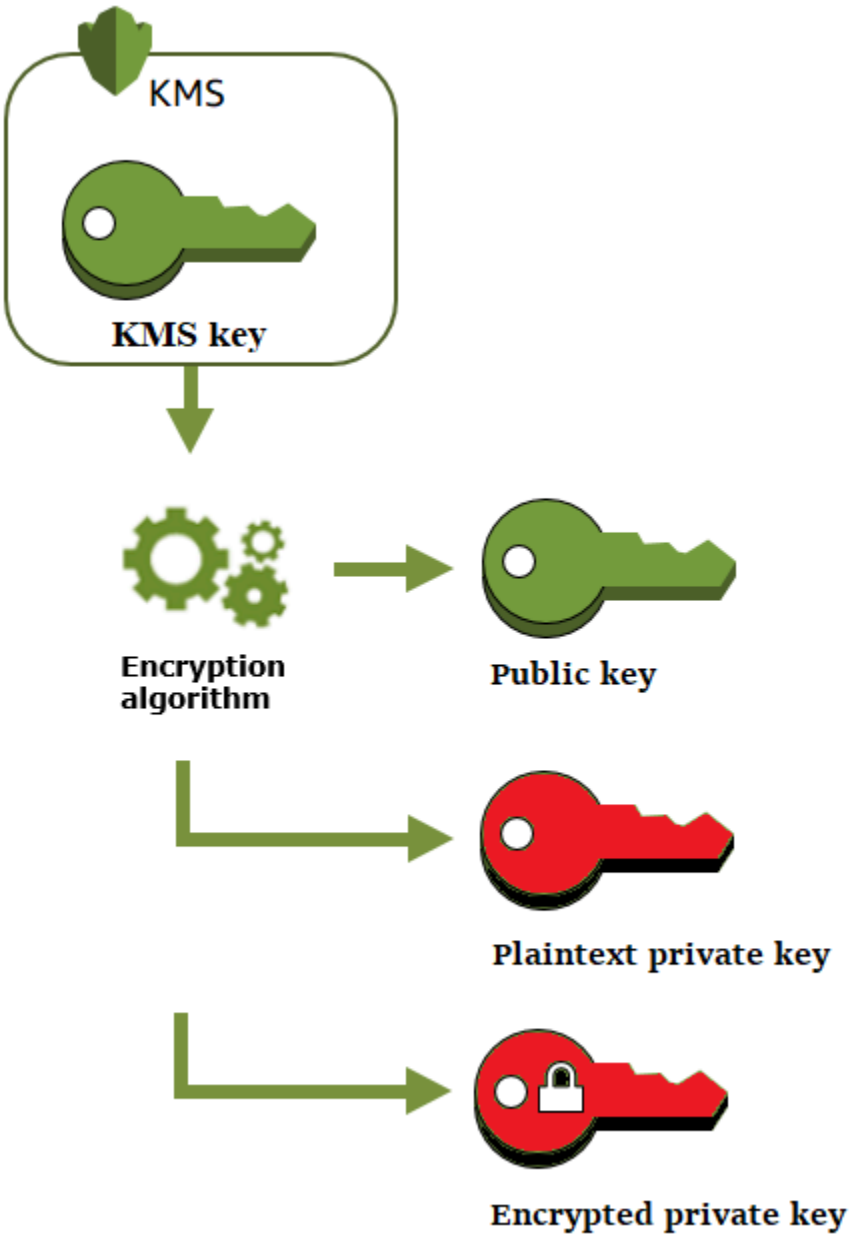
Création d'une paire de clés de données

Pour créer une paire de clés de données, appelez les GenerateDataKeyPairWithoutPlaintextopérations GenerateDataKeyPairor. Spécifiez la clé KMS de chiffrement symétrique que vous souhaitez utiliser pour chiffrer la clé privée.

`GenerateDataKeyPair` renvoie une clé publique en texte brut, une clé privée en texte brut et une clé privée chiffrée. Utilisez cette opération lorsque vous avez besoin immédiatement d'une clé privée en texte brut, par exemple pour générer une signature numérique.

`GenerateDataKeyPairWithoutPlaintext` renvoie une clé publique en texte brut et une clé privée chiffrée, mais pas une clé privée en texte brut. Utilisez cette opération lorsque vous n'avez pas besoin immédiatement d'une clé privée en texte brut, par exemple lorsque vous chiffrez avec une clé publique. Plus tard, lorsque vous avez besoin d'une clé privée en texte brut pour déchiffrer les données, vous pouvez appeler l'opération Decrypt (Déchiffrer).

L'image suivante illustre l'opération `GenerateDataKeyPair`. L'opération `GenerateDataKeyPairWithoutPlaintext` omet la clé privée en texte brut.



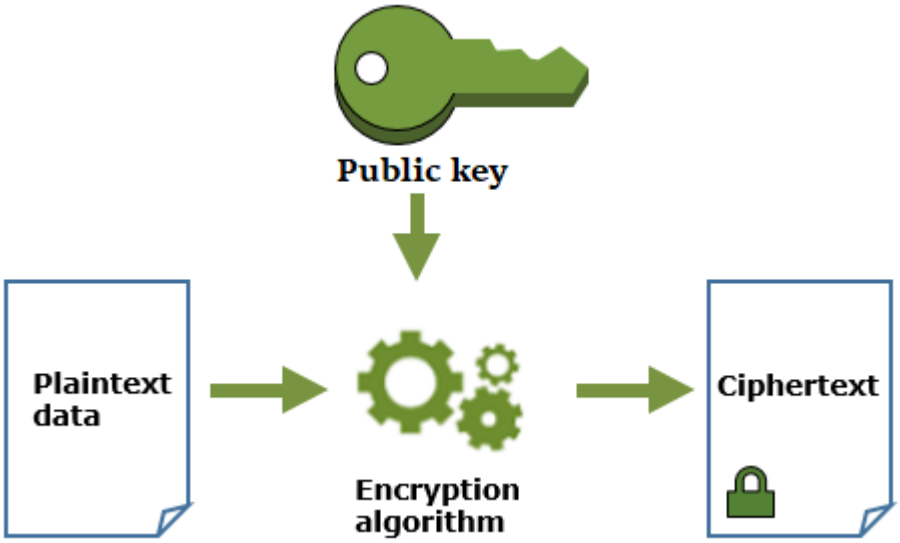
Comment fonctionnent les opérations cryptographiques avec des paires de clés de données

Les rubriques suivantes expliquent les opérations cryptographiques que vous pouvez effectuer avec les paires de clés de données générées par une GenerateDataKeyPairWithoutPlaintextopération GenerateDataKeyPairor et leur fonctionnement.

Chiffrer des données avec une paire de clés de données

Lorsque vous chiffrez avec une paire de clés de données, vous utilisez la clé publique de la paire pour chiffrer les données et la clé privée de la même paire pour déchiffrer les données. Généralement, les paires de clés de données sont utilisées lorsque de nombreuses parties ont besoin de chiffrer des données que seule la partie qui détient la clé privée peut déchiffrer.

Les parties disposant de la clé publique utilisent cette clé pour chiffrer les données, comme indiqué dans le diagramme suivant.

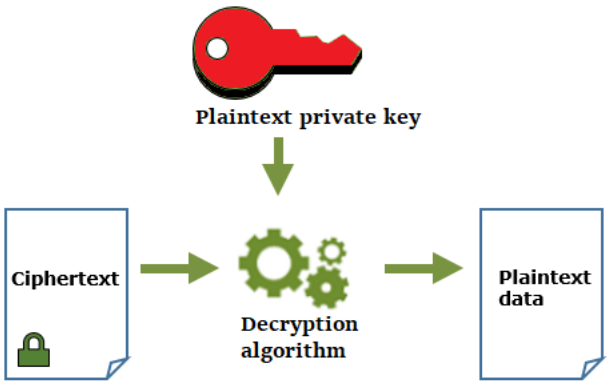


Déchiffrer des données avec une paire de clés de données

Pour déchiffrer vos données, utilisez la clé privée de la paire de clés de données. Pour que l'opération réussisse, les clés publiques et privées doivent être issues de la même paire de clés de données et vous devez utiliser le même algorithme de chiffrement.

Pour déchiffrer la clé privée chiffrée, transmettez-la à l'opération Decrypt (Déchiffrer). Utilisez la clé privée en texte brut pour déchiffrer les données. Ensuite, retirez la clé privée en texte brut de la mémoire dès que possible.

Le diagramme suivant montre comment utiliser la clé privée dans une paire de clés de données pour déchiffrer le texte chiffré.



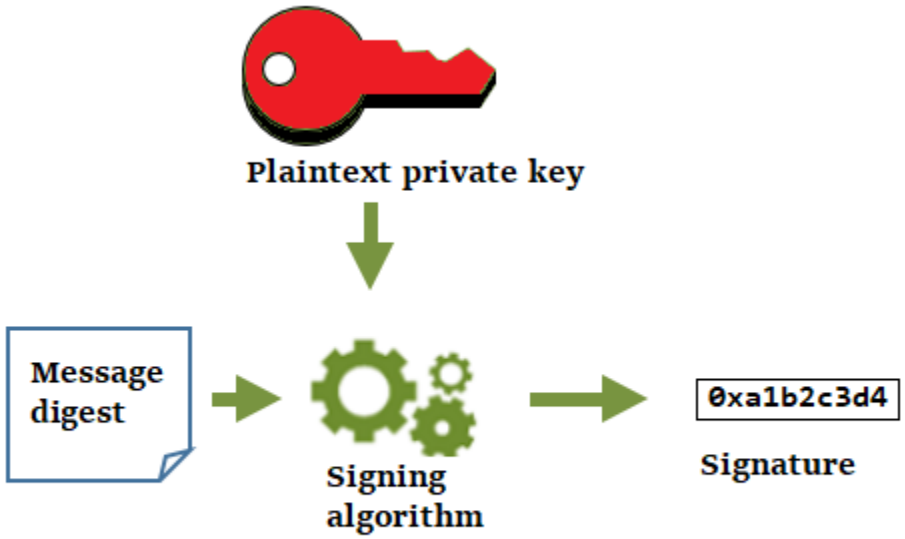
Signer des messages avec une paire de clés de données

Pour générer une signature de chiffrement pour un message, utilisez la clé privée dans la paire de clés de données. Toute personne disposant de la clé publique peut l'utiliser pour vérifier que le message a été signé avec votre clé privée et qu'il n'a pas changé depuis qu'il a été signé.

Si vous chiffrez votre clé privée, transmettez-la à l'opération de déchiffrement. AWS KMS utilise votre clé KMS pour déchiffrer la clé de données, puis renvoie la clé privée en texte brut. Utilisez la clé privée en texte brut pour générer la signature. Ensuite, retirez la clé privée en texte brut de la mémoire dès que possible.

Pour signer un message, créez un résumé de message à l'aide d'une fonction de hachage de chiffrement, telle que la commande `dgst` dans OpenSSL. Ensuite, passez votre clé privée en texte brut à l'algorithme de signature. Le résultat est une signature qui représente le contenu du message. (Vous pourriez être en mesure de signer des messages plus courts sans créer d'abord un résumé. La taille maximale du message varie en fonction de l'outil de signature que vous utilisez.)

Le diagramme suivant montre comment utiliser la clé privée dans une paire de clés de données pour signer un message.

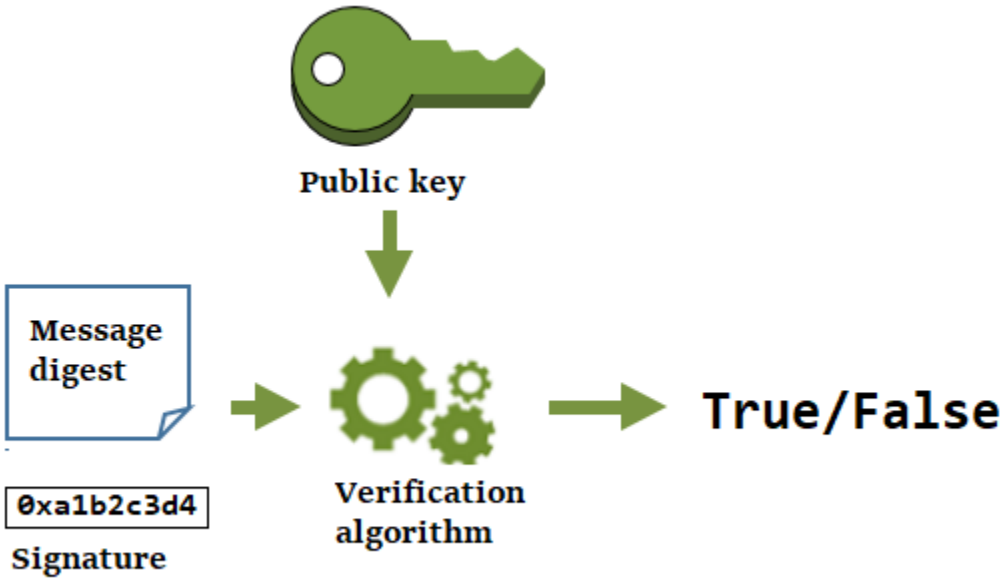


Vérifier une signature avec une paire de clés de données

Toute personne disposant de la clé publique dans votre paire de clés de données peut l'utiliser pour vérifier la signature que vous avez générée avec votre clé privée. La vérification confirme qu'un utilisateur autorisé a signé le message avec la clé privée et l'algorithme de signature spécifiés, et que le message n'a pas changé depuis sa signature.

Pour réussir, la partie qui vérifie la signature doit générer le même type de résumé, utiliser le même algorithme et utiliser la clé publique qui correspond à la clé privée utilisée pour signer le message.

Le diagramme suivant montre comment utiliser la clé publique dans une paire de clés de données pour vérifier une signature de message.



Déterminez un secret partagé à l'aide de paires de clés de données

L'accord clé permet à deux pairs, chacun possédant une paire de clés publique-privée à courbe elliptique, d'établir un secret partagé sur un canal non sécurisé. Pour obtenir un secret partagé, les deux pairs doivent échanger leurs clés publiques via un canal de communication non sécurisé (comme Internet). Ensuite, chaque partie utilise sa clé privée et la clé publique de son homologue pour calculer le même secret partagé à l'aide d'un algorithme d'accord de clé. Vous pouvez utiliser la valeur secrète partagée pour obtenir une clé symétrique capable de chiffrer et de déchiffrer les données envoyées entre les deux homologues, ou de les générer et de les vérifier. HMACs

> **Note**
>
> AWS KMS recommande vivement de vérifier que la clé publique que vous recevez provient de la partie attendue avant de l'utiliser pour dériver un secret partagé.

Effectuez des opérations hors ligne avec des clés publiques

Dans une clé KMS asymétrique, la clé privée est créée AWS KMS et ne sort jamais AWS KMS non chiffrée. Pour utiliser la clé privée, vous devez appeler AWS KMS. Vous pouvez utiliser la clé publique qu'elle contient AWS KMS en appelant les opérations de l' AWS KMS API. Vous pouvez également télécharger la clé publique et la partager pour une utilisation en dehors de AWS KMS.

Vous pouvez partager une clé publique pour permettre à d'autres personnes de AWS KMS chiffrer des données. Vous ne pouvez déchiffrer des données qu'avec votre clé privée. Ou pour permettre à d'autres personnes de vérifier une signature numérique en dehors de AWS KMS que vous avez générée avec votre clé privée. Ou encore, pour partager votre clé publique avec un pair afin d'en déduire un secret partagé.

Lorsque vous utilisez la clé publique contenue dans votre clé KMS asymétrique AWS KMS, vous bénéficiez de l'authentification, de l'autorisation et de la journalisation qui font partie de chaque AWS KMS opération. Vous réduisez également le risque de chiffrement des données qui ne peuvent pas être déchiffrées. Ces fonctionnalités ne sont pas efficaces en dehors de AWS KMS. Pour en savoir plus, consultez Considérations particulières pour le téléchargement de clés publiques.

> **Tip**
>
> Vous recherchez des clés de données ou des clés SSH ? Cette rubrique explique comment gérer les clés asymétriques dans AWS Key Management Service, où la clé privée n'est pas exportable. Pour les paires de clés de données exportables dans lesquelles la clé privée est protégée par une clé KMS de chiffrement symétrique, voir. GenerateDataKeyPair Pour obtenir de l'aide sur le téléchargement de la clé publique associée à une EC2 instance Amazon, consultez la section Extraction de la clé publique dans le guide de l' EC2 utilisateur Amazon et le guide de EC2 l'utilisateur Amazon.

Rubriques

Considérations particulières pour le téléchargement de clés publiques

Pour protéger vos clés KMS, AWS KMS fournit des contrôles d'accès, un chiffrement authentifié et des journaux détaillés de chaque opération. AWS KMS vous permet également d'empêcher l'utilisation des clés KMS, de manière temporaire ou permanente. Enfin, les AWS KMS opérations sont conçues pour minimiser le risque de chiffrer des données qui ne peuvent pas être déchiffrées. Ces fonctionnalités ne sont pas disponibles lorsque vous utilisez des clés publiques téléchargées en dehors de AWS KMS.

Autorisation

Les politiques clés et les politiques IAM qui contrôlent l'accès à la clé KMS interne n' AWS KMS ont aucun effet sur les opérations effectuées en dehors de AWS. Tout utilisateur qui peut obtenir la clé publique peut l'utiliser en dehors de l'UE, AWS KMS même s'il n'est pas autorisé à chiffrer les données ou à vérifier les signatures avec la clé KMS.

Restrictions liées à l'utilisation de la clé

Les principales restrictions d'utilisation ne sont pas applicables en dehors de AWS KMS. Si vous appelez l'opération Encrypt avec une clé KMS comportant le caractère `KeyUsage` de`SIGN_VERIFY`, l' AWS KMS opération échoue. Mais si vous chiffrez des données en dehors de l'extérieur AWS KMS avec une clé publique à partir d'une clé KMS avec un `KeyUsage` `SIGN_VERIFY` ou`KEY_AGREEMENT`, les données ne peuvent pas être déchiffrées.

Restrictions de l'algorithme

Les restrictions relatives aux algorithmes de chiffrement et de signature pris AWS KMS en charge ne sont pas efficaces en dehors de AWS KMS. Si vous chiffrez des données avec la clé publique à partir d'une clé KMS extérieure et que AWS KMS vous utilisez un algorithme de AWS KMS chiffrement non compatible, les données ne peuvent pas être déchiffrées.

Désactivation et suppression des clés KMS

Les mesures que vous pouvez prendre pour empêcher l'utilisation de la clé KMS dans le cadre d'une opération cryptographique AWS KMS interne n'empêchent personne d'utiliser la clé publique en dehors de AWS KMS. Par exemple, la désactivation d'une clé KMS, la planification de la suppression d'une clé KMS, la suppression d'une clé KMS ou la suppression des éléments d'une clé KMS n'ont aucun effet sur une clé publique en dehors de AWS KMS. Si vous supprimez une clé KMS asymétrique ou si vous supprimez ou perdez son contenu clé, les données que vous chiffrez avec une clé publique extérieure AWS KMS sont irrécupérables.

Journalisation

AWS CloudTrail les journaux qui enregistrent chaque AWS KMS opération, y compris la demande, la réponse, la date, l'heure et l'utilisateur autorisé, n'enregistrent pas l'utilisation de la clé publique en dehors de AWS KMS.

Vérification hors ligne à l'aide de paires de SM2 clés (régions de Chine uniquement)

Pour vérifier une signature en dehors AWS KMS d'une clé SM2 publique, vous devez spécifier l'identifiant distinctif. Par défaut, AWS KMS utilise `1234567812345678` comme identifiant distinctif. Pour plus d'informations, consultez la section Vérification hors ligne à l'aide de paires de SM2 clés (régions de Chine uniquement).

Télécharger la clé publique

Vous pouvez télécharger la clé publique à partir d'une paire de clés KMS asymétriques dans la AWS KMS console ou en utilisant l'GetPublicKeyopération. Pour télécharger la clé publique, vous devez disposer d'une `kms:GetPublicKey` autorisation sur la clé KMS asymétrique.

La clé publique AWS KMS renvoyée est une clé publique X.509 codée DER, également connue sous le nom de `SubjectPublicKeyInfo` (SPKI), telle que définie dans la RFC 5280. Lorsque vous utilisez l'API HTTP ou le AWS CLI, la valeur est codée en Base64. Dans le cas contraire, il n'est pas codé en Base64.

Pour télécharger la clé publique à partir d'une paire de clés KMS asymétriques, vous devez `kms:GetPublicKey` disposer d'autorisations. Pour plus d'informations sur AWS KMS les autorisations, consultez leRéférence des autorisations .

Utilisation de la AWS KMS console

Vous pouvez utiliser le AWS Management Console pour afficher, copier et télécharger la clé publique à partir d'une clé KMS asymétrique présente dans votre Compte AWS. Pour télécharger la clé publique à partir d'une clé KMS asymétrique dans un autre format Compte AWS, utilisez l' AWS KMS API.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.

3. Dans le volet de navigation, sélectionnez Clés gérées par le client.
4. Choisissez l'alias ou l'ID de clé d'une clé KMS asymétrique.
5. Choisissez l'onglet Cryptographic configuration (Configuration de chiffrement). Enregistrez les valeurs des champs Spécifications de la clé, Utilisation de la clé et Algorithmes de chiffrement ou Algorithmes de signature. Vous devez utiliser ces valeurs pour utiliser la clé publique en dehors de AWS KMS. Assurez-vous de partager ces informations lorsque vous partagez la clé publique.
6. Choisissez l'onglet Clé publique.
7. Pour copier la clé publique dans le presse-papiers, choisissez Copier. Pour télécharger la clé publique dans un fichier, choisissez Télécharger.

Utilisation de l' AWS KMS API

L'GetPublicKeyopération renvoie la clé publique sous forme de clé KMS asymétrique. Il renvoie également des informations critiques dont vous avez besoin pour utiliser correctement la clé publique en dehors de celles-ci AWS KMS, notamment l'utilisation des clés et les algorithmes de chiffrement. Veillez à enregistrer ces valeurs et à les partager chaque fois que vous partagez la clé publique.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Pour identifier une clé KMS, utilisez son ID de clé, son ARN de clé, son nom d'alias ou son ARN d'alias. Lorsque vous utilisez un nom d'alias, préfixez-le avec alias/. Pour spécifier une clé KMS dans une autre Compte AWS, vous devez utiliser son ARN de clé ou son alias ARN.

Avant d'exécuter cette commande, remplacez l'exemple de nom d'alias par un identifiant valide pour la clé KMS. Pour exécuter cette commande, vous devez disposer `kms:GetPublicKey` d'autorisations sur la clé KMS.

```
$ aws kms get-public-key --key-id alias/example_RSA_3072

{
    "KeySpec": "RSA_3072",
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "EncryptionAlgorithms": [
        "RSAES_OAEP_SHA_1",
        "RSAES_OAEP_SHA_256"
```

```
    ],
    "PublicKey": "MIIBojANBgkqhkiG..."
}
```

Exemples d'opérations hors ligne

Après avoir téléchargé la clé publique de votre paire de clés KMS asymétriques, vous pouvez la partager avec d'autres utilisateurs et l'utiliser pour effectuer des opérations hors ligne.

AWS CloudTrail les journaux qui enregistrent chaque AWS KMS opération, y compris la demande, la réponse, la date, l'heure et l'utilisateur autorisé, n'enregistrent pas l'utilisation de la clé publique en dehors de AWS KMS.

Cette rubrique fournit des exemples d'opérations hors ligne et des détails fournis par les outils AWS KMS pour faciliter les opérations hors ligne.

Rubriques

- Découverte de secrets partagés hors ligne
- Vérification hors ligne avec des paires de clés ML-DSA
- Vérification hors ligne à l'aide de paires de SM2 clés (régions de Chine uniquement)

Découverte de secrets partagés hors ligne

Vous pouvez télécharger la clé publique de votre paire de clés ECC pour l'utiliser dans des opérations hors ligne, c'est-à-dire des opérations en dehors de AWS KMS.

La procédure pas à pas suivante montre une méthode permettant de dériver un secret partagé en dehors de l'utilisation de la clé publique d'une paire AWS KMS de clés ECC KMS et d'une clé privée créée avec OpenSSL.

1. Créez une paire de clés ECC dans OpenSSL et préparez-la pour une utilisation avec. AWS KMS

```
// Create an ECC key pair in OpenSSL and save the private key in
 openssl_ecc_key_priv.pem
export OPENSSL_CURVE_NAME="P-256"
export KMS_CURVE_NAME="ECC_NIST_P256"

export OPENSSL_KEY1_PRIV_PEM="openssl_ecc_key1_priv.pem"
```

```
openssl ecparam -name ${OPENSSL_CURVE_NAME} -genkey -out ${OPENSSL_KEY1_PRIV_PEM}


// Derive the public key from the private key
export OPENSSL_KEY1_PUB_PEM="openssl_ecc_key1_pub.pem"
openssl ec -in ${OPENSSL_KEY1_PRIV_PEM} -pubout -outform pem \
    -out ${OPENSSL_KEY1_PUB_PEM}

// View the PEM file containing the public key and extract the public key as a
// Base64 encoded string into OPENSSL_KEY1_PUB_BASE64 for use with AWS KMS
export OPENSSL_KEY1_PUB_BASE64=`cat ${OPENSSL_KEY1_PUB_PEM} | \
    tee /dev/stderr | grep -v "PUBLIC KEY" | tr -d "\n"`
```

2. Créez une paire de clés d'accord de clés ECC AWS KMS et préparez-la pour une utilisation avec OpenSSL.

```
// Create a KMS key on the same curve as the key pair from step 1
// with a key usage of KEY_AGREEMENT
// Save its ARN in KMS_KEY1_ARN.
export KMS_KEY1_ARN=`aws kms create-key --key-spec ${KMS_CURVE_NAME} \
    --key-usage KEY_AGREEMENT | tee /dev/stderr | jq -r .KeyMetadata.Arn`

// Download the public key and save the Base64-encoded version in KMS_KEY1_PUB_BASE64

export KMS_KEY1_PUB_BASE64=`aws kms get-public-key --key-id ${KMS_KEY1_ARN} | \
    tee /dev/stderr | jq -r .PublicKey`

// Create a PEM file for the public KMS key for use with OpenSSL
export KMS_KEY1_PUB_PEM="aws_kms_ecdh_key1_pub.pem"
echo "-----BEGIN PUBLIC KEY-----" > ${KMS_KEY1_PUB_PEM}
echo ${KMS_KEY1_PUB_BASE64} | fold -w 64 >> ${KMS_KEY1_PUB_PEM}
echo "-----END PUBLIC KEY-----" >> ${KMS_KEY1_PUB_PEM}
```

3. Déterminez le secret partagé dans OpenSSL à l'aide de la clé privée d'OpenSSL et de la clé KMS publique.

```
export OPENSSL_SHARED_SECRET1_BIN="openssl_shared_secret1.bin"
openssl pkeyutl -derive -inkey ${OPENSSL_KEY1_PRIV_PEM} \
    -peerkey ${KMS_KEY1_PUB_PEM} -out ${OPENSSL_SHARED_SECRET1_BIN}
```

Vérification hors ligne avec des paires de clés ML-DSA

AWS KMS prend en charge une variante couverte de la signature ML-DSA, telle que décrite dans la section 3.4 de la norme Federal Information Processing Standards (FIPS) 204 pour les messages d'une taille maximale de 4 Ko d'octets.

Pour signer des messages d'une taille supérieure à 4 Ko, vous devez effectuer l'étape de prétraitement des messages en dehors de AWS KMS. Cette étape de hachage crée un message de 64 octets représentatif de μ, tel que défini dans le NIST FIPS 204, section 6.2.

AWS KMS possède un type de message appelé `EXTERNAL_MU` pour les messages supérieurs à 4 Ko. Lorsque vous l'utilisez à la place du type de `RAW` message, AWS KMS :

- Suppose que vous avez déjà effectué l'étape de hachage
- Ignore son processus de hachage interne
- Fonctionne avec des messages de toutes tailles

Lorsque vous vérifiez un message, la méthode que vous utilisez dépend de la restriction de taille du système ou de la bibliothèque externe et de la prise en charge du message de 64 octets représentant μ :

- Si le message est inférieur à la limite de taille, utilisez le type de `RAW` message.
- Si le message est supérieur à la limite de taille, utilisez le μ représentatif dans le système externe.

Les sections suivantes montrent comment signer AWS KMS et vérifier des messages à l'aide d'OpenSSL. Nous fournissons des exemples de messages inférieurs ou supérieurs à la limite de 4 Ko imposée par AWS KMS. OpenSSL n'impose pas de limite à la taille des messages à des fins de vérification.

Pour les deux exemples, vous devez d'abord obtenir la clé publique auprès de AWS KMS. Utilisez la commande AWS CLI suivante :

```
aws kms get-public-key \
    --key-id _<1234abcd-12ab-34cd-56ef-1234567890ab>_ \
    --output text \
    --query PublicKey | base64 --decode > public_key.der
```

Taille du message inférieure à 4 Ko

Pour les messages de moins de 4 Ko, utilisez le type de `RAW` message avec AWS KMS. Bien que vous puissiez l'utiliser`EXTERNAL_MU`, cela n'est pas nécessaire pour les messages dont la taille est inférieure à la limite de taille.

Utilisez la AWS CLI commande suivante pour signer le message :

```
aws kms sign \
    --key-id _<1234abcd-12ab-34cd-56ef-1234567890ab>_ \
    --message 'your message' \
    --message-type RAW \
    --signing-algorithm ML_DSA_SHAKE_256 \
    --output text \
    --query Signature | base64 --decode > ExampleSignature.bin
```

Pour vérifier ce message à l'aide d'OpenSSL, utilisez la commande suivante :

```
echo -n 'your message' | ./openssl dgst -verify public_key.der -signature
 ExampleSignature.bin
```

Taille du message supérieure à 4 Ko

Pour signer des messages de plus de 4 Ko, utilisez le type de `EXTERNAL_MU` message. Lorsque vous l'utilisez`EXTERNAL_MU`, vous pré-hachez le message en externe en un μ représentatif de 64 octets, tel que défini dans la section 6.2 du NIST FIPS 204, et vous le transmettez aux opérations de signature ou de vérification. Notez que cela est différent du « MLDSA pré-hachage » ou du HashML-DSA définis dans la section 5.4 du NIST FIPS 204.

1. Commencez par créer un préfixe de message. Le préfixe contient un séparateur de domaine, la longueur de tout contexte et le contexte. La valeur par défaut pour le séparateur de domaine et la longueur du contexte est zéro.
2. Ajoutez le préfixe du message au message.
3. SHAKE256 À utiliser pour hacher la clé publique et l'ajouter au résultat de l'étape 2.
4. Enfin, hachez le résultat de l'étape 3 pour produire un 64 `EXTERNAL_MU` octets.

L'exemple suivant utilise OpenSSL 3.5 pour créer : `EXTERNAL_MU`

```
{
    openssl asn1parse -inform DER -in public_key.der -strparse 17 -noout -out - 2>/dev/
null |
    openssl dgst -provider default -shake256 -xoflen 64 -binary;
    printf '\x00\x00';
    echo -n "your message"
} | openssl dgst -provider default -shake256 -xoflen 64 -binary > mu.bin
```

Après avoir créé le `mu.bin` fichier, appelez l' AWS KMS API avec la commande suivante pour signer le message :

```
aws kms sign \
    --key-id _<1234abcd-12ab-34cd-56ef-1234567890ab>_ \
    --message fileb://mu.bin \
    --message-type EXTERNAL_MU \
    --signing-algorithm ML_DSA_SHAKE_256 \
    --output text \
    --query Signature | base64 --decode > ExampleSignature.bin
```

La signature qui en résulte est identique à `RAW` celle du message d'origine. Vous pouvez utiliser la même commande OpenSSL 3.5 pour vérifier le message :

```
echo -n 'your message' | ./openssl dgst -verify public_key.der -signature
 ExampleSignature.bin
```

Vérification hors ligne à l'aide de paires de SM2 clés (régions de Chine uniquement)

Pour vérifier une signature en dehors AWS KMS d'une clé SM2 publique, vous devez spécifier l'identifiant distinctif. Lorsque vous transmettez un message brut à l'API Sign, il AWS KMS utilise l'identifiant distinctif par défaut`1234567812345678`, défini par l'OSCA en 0009-2012. `MessageType:RAW` GM/T Vous ne pouvez pas spécifier votre propre ID distinctif dans AWS KMS.

Toutefois, si vous générez un résumé de message en dehors de AWS, vous pouvez spécifier votre propre identifiant distinctif, puis transmettre le résumé du message, AWS KMS à `MessageType:DIGEST`, pour le signer. Pour ce faire, modifiez le paramètre `DEFAULT_DISTINGUISHING_ID` valeur dans la clé `SM2OfflineOperationHelper` classe. L'ID distinctif que vous spécifiez peut être n'importe quelle chaîne de 8 192 caractères maximum. Après

avoir AWS KMS signé le résumé du message, vous avez besoin soit du résumé du message, soit du message et de l'ID distinctif utilisés pour calculer le résumé afin de le vérifier hors ligne.

> **Important**
>
> Le `SM2OfflineOperationHelper` code de référence est conçu pour être compatible avec Bouncy Castle version 1.68. Pour obtenir de l'aide sur les autres versions, contactezbouncycastle.org.

classe **`SM2OfflineOperationHelper`**

Pour vous aider à effectuer des opérations hors ligne avec des SM2 clés, la `SM2OfflineOperationHelper` classe pour Java possède des méthodes qui exécutent les tâches à votre place. Vous pouvez utiliser cette classe d'assistance comme modèle pour d'autres fournisseurs de cryptographie.

Dans AWS KMS ce cadre, les conversions de texte chiffré brut et les calculs du résumé des messages SM2 DSA sont effectués automatiquement. Tous les fournisseurs de cryptographie ne mettent pas SM2 en œuvre de la même manière. Certaines bibliothèques, comme les versions 1.1.1 et ultérieures d'OpenSSL, exécutent ces actions automatiquement. AWS KMS a confirmé ce comportement lors de tests avec la version 3.0 d'OpenSSL. Utilisez les classes `SM2OfflineOperationHelper` avec des bibliothèques, comme Bouncy Castle, qui vous obligent à effectuer ces conversions et ces calculs manuellement.

La classe `SM2OfflineOperationHelper` fournit des méthodes pour les opérations hors ligne suivantes :

- Calcul de l'algorithme Message Digest

  Pour générer un résumé de message hors ligne que vous pouvez utiliser pour la vérification hors ligne ou que vous pouvez transmettre AWS KMS pour le signer, utilisez `calculateSM2Digest` cette méthode. Le `calculateSM2Digest` procédé génère un condensé de message à l'aide de l'algorithme SM3 de hachage. L'GetPublicKeyAPI renvoie votre clé publique au format binaire. Vous devez analyser la clé binaire dans un Java PublicKey. Fournissez la clé publique analysée avec le message. La méthode combine automatiquement votre message avec l'identifiant distinctif par défaut, `1234567812345678`, mais vous pouvez définir votre propre ID distinctif en modifiant la valeur `DEFAULT_DISTINGUISHING_ID`.

- Vérification

  Pour vérifier une signature hors ligne, utilisez la méthode `offlineSM2DSAVerify`. La méthode `offlineSM2DSAVerify` utilise le résumé du message calculé à partir de l'ID distinctif spécifié et du message d'origine que vous avez fourni pour vérifier la signature numérique. L'GetPublicKeyAPI renvoie votre clé publique au format binaire. Vous devez analyser la clé binaire dans un Java PublicKey. Fournissez la clé publique analysée avec le message d'origine et la signature que vous souhaitez vérifier. Pour plus de détails, consultez la section Vérification hors ligne à l'aide de paires de SM2 clés.

- Chiffrement

  Pour chiffrer du texte en clair hors ligne, utilisez la méthode `offlineSM2PKEEncrypt`. Cette méthode garantit que le texte chiffré est dans un format AWS KMS déchiffrable. Le `offlineSM2PKEEncrypt` procédé chiffre le texte en clair, puis convertit le texte chiffré brut produit par SM2 PKE au format ASN.1. L'GetPublicKeyAPI renvoie votre clé publique au format binaire. Vous devez analyser la clé binaire dans un Java PublicKey. Fournissez la clé publique analysée avec le texte brut que vous souhaitez chiffrer.

  Si vous n'êtes pas sûr(e) de devoir effectuer la conversion, utilisez l'opération OpenSSL suivante pour tester le format de votre texte chiffré. Si l'opération échoue, vous devez convertir le texte chiffré au format ASN.1.

  ```
  openssl asn1parse -inform DER -in ciphertext.der
  ```

Par défaut, la `SM2OfflineOperationHelper` classe utilise l'ID distinctif par défaut lors de la génération de résumés de messages pour les opérations SM2 DSA. `1234567812345678`

```
package com.amazon.kms.utils;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import java.io.IOException;
import java.math.BigInteger;
import java.nio.ByteBuffer;
import java.nio.charset.StandardCharsets;
import java.security.InvalidKeyException;
import java.security.MessageDigest;
```

```
import java.security.NoSuchAlgorithmException;
import java.security.NoSuchProviderException;
import java.security.PrivateKey;
import java.security.PublicKey;

import org.bouncycastle.crypto.CryptoException;
import org.bouncycastle.jce.interfaces.ECPublicKey;

import java.util.Arrays;

import org.bouncycastle.asn1.ASN1EncodableVector;
import org.bouncycastle.asn1.ASN1Integer;
import org.bouncycastle.asn1.DEROctetString;
import org.bouncycastle.asn1.DERSequence;
import org.bouncycastle.asn1.gm.GMNamedCurves;
import org.bouncycastle.asn1.x9.X9ECParameters;
import org.bouncycastle.crypto.CipherParameters;
import org.bouncycastle.crypto.params.ParametersWithID;
import org.bouncycastle.crypto.params.ParametersWithRandom;
import org.bouncycastle.crypto.signers.SM2Signer;
import org.bouncycastle.jcajce.provider.asymmetric.util.ECUtil;

public class SM2OfflineOperationHelper {
    // You can change the DEFAULT_DISTINGUISHING_ID value to set your own
 distinguishing ID,
    // the DEFAULT_DISTINGUISHING_ID can be any string up to 8,192 characters long.
    private static final byte[] DEFAULT_DISTINGUISHING_ID =
 "1234567812345678".getBytes(StandardCharsets.UTF_8);
    private static final X9ECParameters SM2_X9EC_PARAMETERS =
 GMNamedCurves.getByName("sm2p256v1");

    // ***calculateSM2Digest***
    // Calculate message digest
    public static byte[] calculateSM2Digest(final PublicKey publicKey, final byte[]
 message) throws
            NoSuchProviderException, NoSuchAlgorithmException {
        final ECPublicKey ecPublicKey = (ECPublicKey) publicKey;

        // Generate SM3 hash of default distinguishing ID, 1234567812345678
        final int entlenA = DEFAULT_DISTINGUISHING_ID.length * 8;
        final byte [] entla = new byte[] { (byte) (entlenA & 0xFF00), (byte) (entlenA &
 0x00FF) };
        final byte [] a = SM2_X9EC_PARAMETERS.getCurve().getA().getEncoded();
        final byte [] b = SM2_X9EC_PARAMETERS.getCurve().getB().getEncoded();
```

```
        final byte [] xg = SM2_X9EC_PARAMETERS.getG().getXCoord().getEncoded();
        final byte [] yg = SM2_X9EC_PARAMETERS.getG().getYCoord().getEncoded();
        final byte[] xa = ecPublicKey.getQ().getXCoord().getEncoded();
        final byte[] ya = ecPublicKey.getQ().getYCoord().getEncoded();
        final byte[] za = MessageDigest.getInstance("SM3", "BC")
                .digest(ByteBuffer.allocate(entla.length +
DEFAULT_DISTINGUISHING_ID.length + a.length + b.length + xg.length + yg.length +
                        xa.length +
ya.length).put(entla).put(DEFAULT_DISTINGUISHING_ID).put(a).put(b).put(xg).put(yg).put(xa).put
                        .array());

        // Combine hashed distinguishing ID with original message to generate final
digest
        return MessageDigest.getInstance("SM3", "BC")
                .digest(ByteBuffer.allocate(za.length +
message.length).put(za).put(message)
                        .array());
    }

    // ***offlineSM2DSAVerify***
    // Verify digital signature with SM2 public key
    public static boolean offlineSM2DSAVerify(final PublicKey publicKey, final byte []
message,
            final byte [] signature) throws InvalidKeyException {
        final SM2Signer signer = new SM2Signer();
        CipherParameters cipherParameters =
ECUtil.generatePublicKeyParameter(publicKey);
        cipherParameters = new ParametersWithID(cipherParameters,
DEFAULT_DISTINGUISHING_ID);
        signer.init(false, cipherParameters);
        signer.update(message, 0, message.length);
        return signer.verifySignature(signature);
    }

    // ***offlineSM2PKEEncrypt***
    // Encrypt data with SM2 public key
    public static byte[] offlineSM2PKEEncrypt(final PublicKey publicKey, final byte []
plaintext) throws
            NoSuchPaddingException, NoSuchAlgorithmException, NoSuchProviderException,
InvalidKeyException,
            BadPaddingException, IllegalBlockSizeException, IOException {
        final Cipher sm2Cipher = Cipher.getInstance("SM2", "BC");
        sm2Cipher.init(Cipher.ENCRYPT_MODE, publicKey);
```

```
        // By default, Bouncy Castle returns raw ciphertext in the c1c2c3 format
        final byte [] cipherText = sm2Cipher.doFinal(plaintext);

        // Convert the raw ciphertext to the ASN.1 format before passing it to AWS KMS
        final ASN1EncodableVector asn1EncodableVector = new ASN1EncodableVector();
        final int coordinateLength = (SM2_X9EC_PARAMETERS.getCurve().getFieldSize() +
 7) / 8 * 2 + 1;
        final int sm3HashLength = 32;
        final int xCoordinateInCipherText = 33;
        final int yCoordinateInCipherText = 65;
        byte[] coords = new byte[coordinateLength];
        byte[] sm3Hash = new byte[sm3HashLength];
        byte[] remainingCipherText = new byte[cipherText.length - coordinateLength -
 sm3HashLength];

        // Split components out of the ciphertext
        System.arraycopy(cipherText, 0, coords, 0, coordinateLength);
        System.arraycopy(cipherText, cipherText.length - sm3HashLength, sm3Hash, 0,
 sm3HashLength);
        System.arraycopy(cipherText, coordinateLength, remainingCipherText,
 0,cipherText.length - coordinateLength - sm3HashLength);

        // Build standard SM2PKE ASN.1 ciphertext vector
        asn1EncodableVector.add(new ASN1Integer(new BigInteger(1,
 Arrays.copyOfRange(coords, 1, xCoordinateInCipherText))));
        asn1EncodableVector.add(new ASN1Integer(new BigInteger(1,
 Arrays.copyOfRange(coords, xCoordinateInCipherText, yCoordinateInCipherText))));
        asn1EncodableVector.add(new DEROctetString(sm3Hash));
        asn1EncodableVector.add(new DEROctetString(remainingCipherText));

        return new DERSequence(asn1EncodableVector).getEncoded("DER");
    }
}
```

Moniteur AWS KMS keys

La surveillance joue un rôle important dans la compréhension de la disponibilité, de l'état et de l'utilisation de vos solutions, AWS KMS ainsi que AWS KMS keys dans le maintien de la fiabilité, de la disponibilité et des performances de vos AWS solutions. La collecte de données de surveillance à partir de toutes les parties de votre solution AWS vous aidera à résoudre des problèmes de défaillance multipoint, le cas échéant. Toutefois, avant de commencer la surveillance de cos clés KMS, créez un plan de surveillance incluant les réponses aux questions suivantes :

- Quels sont les objectifs de la surveillance ?
- Quelles sont les ressources à surveiller ?
- À quelle fréquence les ressources doivent-elles être surveillées ?
- Quels outils de surveillance utiliser ?
- Qui exécute les tâches de supervision ?
- Qui doit être informé en cas de problème ?

L'étape suivante consiste à contrôler vos clés KMS au fil du temps pour établir une référence d'utilisation normale de AWS KMS et une base pour les attentes dans votre environnement. Lorsque vous contrôlez vos clés KMS, conservez les données d'historique de surveillance pour les comparer aux données actuelles afin d'identifier des modèles normaux et des anomalies, et de concevoir des méthodes pour résoudre les problèmes.

Par exemple, vous pouvez surveiller AWS KMS l'activité des API et les événements qui affectent vos clés KMS. Lorsque les données dépassent les normes supérieures ou inférieures que vous avez établies, il peut être nécessaire d'enquêter ou de prendre des mesures correctives.

Pour établir une référence pour les modèles normaux, surveillez les éléments suivants :

- AWS KMS Activité de l'API pour les opérations du plan de données. Il s'agit d'opérations cryptographiques qui utilisent une clé KMS, telles que Decrypt, Encrypt et ReEncrypt. GenerateDataKey
- AWS KMS Activité d'API pour les opérations du plan de contrôle qui sont importantes pour vous. Ces opérations gèrent une clé KMS, et vous souhaiterez peut-être surveiller celles qui modifient la disponibilité d'une clé KMS (telles que ScheduleKeyDeletionCancelKeyDeletion, DisableKey, EnableKey, ImportKeyMaterial, et DeleteImportedKeyMaterial) ou le contrôle d'accès d'une clé KMS (comme PutKeyPolicyet RevokeGrant).

- Autres AWS KMS indicateurs (tels que le temps restant avant l'expiration du contenu clé importé) et événements (tels que l'expiration du contenu clé importé ou la suppression ou la rotation des clés d'une clé KMS).

Outils de surveillance

AWS fournit divers outils que vous pouvez utiliser pour surveiller vos clés KMS. Vous pouvez configurer certains outils pour qu'ils effectuent la supervision automatiquement, tandis que d'autres nécessitent une intervention manuelle. Nous vous recommandons d'automatiser le plus possible les tâches de supervision.

Outils de surveillance automatique

Vous pouvez utiliser les outils de surveillance automatique ci-dessous pour contrôler vos clés KMS et signaler un changement éventuel.

- AWS CloudTrail Surveillance des journaux : partagez les fichiers journaux entre les comptes, surveillez les fichiers CloudTrail CloudWatch journaux en temps réel en les envoyant à Logs, rédigez des applications de traitement des journaux avec la bibliothèque de CloudTrail traitement et vérifiez que vos fichiers journaux n'ont pas changé après leur livraison par CloudTrail. Pour plus d'informations, consultez la section Utilisation des fichiers CloudTrail journaux dans le guide de AWS CloudTrail l'utilisateur.
- Amazon CloudWatch Alarms : surveillez une seule métrique sur une période que vous spécifiez et effectuez une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de périodes. L'action est une notification envoyée à une rubrique Amazon Simple Notification Service (Amazon SNS) ou à une politique Amazon EC2 Auto Scaling. CloudWatch les alarmes n'appellent pas d'actions simplement parce qu'elles sont dans un état particulier ; l'état doit avoir changé et être maintenu pendant un certain nombre de périodes. Pour de plus amples informations, veuillez consulter Surveillez les clés KMS avec Amazon CloudWatch.
- Amazon EventBridge — Associez les événements et acheminez-les vers une ou plusieurs fonctions ou flux cibles afin de capturer des informations d'état et, si nécessaire, d'apporter des modifications ou de prendre des mesures correctives. Pour plus d'informations, consultez Surveillez les clés KMS avec Amazon EventBridge le guide de EventBridge l'utilisateur Amazon.
- Amazon CloudWatch Logs — Surveillez, stockez et accédez à vos fichiers journaux depuis AWS CloudTrail ou d'autres sources. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.

Outils de surveillance manuelle

Un autre élément important de la surveillance des clés KMS consiste à surveiller manuellement les éléments non couverts par les CloudWatch alarmes et les événements. Le tableau de bord AWS KMS CloudWatch AWS Trusted Advisor,, et les autres AWS tableaux de bord fournissent une at-a-glance vue de l'état de votre AWS environnement.

Vous pouvez personnaliser les pages Clés gérées par AWS et les clés gérées par le client de la console AWS KMS pour afficher les informations suivantes sur chaque clé KMS :

- ID de clé
- Statut
- Date de création
- Date d'expiration (pour les clés KMS avec des éléments de clé importés)
- Origin
- ID de magasin de clés personnalisé (pour les clés KMS dans des magasins de clés personnalisés)

Le tableau de bord de la console CloudWatch présente les éléments suivants :

- Alarmes et statuts en cours
- Graphiques des alarmes et des ressources
- Statut d’intégrité du service

En outre, vous pouvez utiliser CloudWatch pour effectuer les opérations suivantes :

- Créer des tableaux de bord personnalisés pour surveiller les services de votre choix
- Représenter graphiquement les données de métriques pour résoudre les problèmes et découvrir les tendances
- Recherchez et parcourez tous les indicateurs de vos AWS ressources
- Créer et modifier des alarmes pour être informé des problèmes

AWS Trusted Advisor peut vous aider à surveiller vos AWS ressources afin d'améliorer les performances, la fiabilité, la sécurité et la rentabilité. Quatre Trusted Advisor chèques sont disponibles pour tous les utilisateurs ; plus de 50 chèques sont disponibles pour les utilisateurs

disposant d'un plan de support Business ou Enterprise. Pour de plus amples informations, veuillez consulter AWS Trusted Advisor.

Journalisation des appels d' AWS KMS API avec AWS CloudTrail

AWS KMS est intégré à AWS CloudTrailun service qui enregistre tous les appels AWS KMS adressés par les utilisateurs, les rôles et les autres AWS services. CloudTrail capture tous les appels d'API AWS KMS sous forme d'événements, y compris les appels depuis la AWS KMS console AWS KMS APIs, les AWS CloudFormation modèles, le AWS Command Line Interface (AWS CLI) et Outils AWS pour PowerShell.

CloudTrail enregistre toutes les AWS KMS opérations, y compris les opérations en lecture seule, telles que ListAliaseset GetKeyRotationStatus, les opérations qui gèrent les clés KMS, telles que CreateKeyet, et les opérations cryptographiques PutKeyPolicy, telles que GenerateDataKeyet Decrypt. Il enregistre également les opérations internes AWS KMS qui vous concernent, telles que DeleteExpiredKeyMaterialDeleteKey, SynchronizeMultiRegionKey, et RotateKey.

CloudTrail enregistre toutes les opérations réussies et, dans certains scénarios, les tentatives d'appels qui ont échoué, par exemple lorsque l'accès à une ressource est refusé à l'appelant. Les opérations intercomptes sur clés KMS sont journalisées à la fois sur le compte appelant et le compte propriétaire de la clé KMS. Toutefois, les AWS KMS demandes entre comptes rejetées parce que l'accès est refusé ne sont enregistrées que dans le compte de l'appelant.

Pour des raisons de sécurité, certains champs sont omis des entrées du AWS KMS journal, tels que le `Plaintext` paramètre d'une demande de chiffrement, la réponse à une opération cryptographique GetKeyPolicyou toute autre opération cryptographique. Pour faciliter la recherche d'entrées de CloudTrail journal pour des clés KMS spécifiques, AWS KMS ajoute l'ARN clé de la clé KMS affectée au `responseElements` champ des entrées de journal pour certaines opérations de gestion des AWS KMS clés, même si l'opération d'API ne renvoie pas l'ARN de la clé.

Bien que, par défaut, toutes les AWS KMS actions soient enregistrées en tant qu' CloudTrail événements, vous pouvez AWS KMS les exclure d'un CloudTrail suivi. Pour plus de détails, consultez Exclure AWS KMS des événements d'un parcours.

En savoir plus :

- Pour CloudTrail obtenir des exemples d' AWS KMS opérations enregistrées pour une enclave AWS Nitro, consultezDemandes de surveillance pour les enclaves Nitro.

Rubriques

Recherche d'entrées de AWS KMS journal dans CloudTrail

Pour rechercher des entrées de CloudTrail journal, utilisez la CloudTrail console ou l'CloudTrail LookupEventsopération. CloudTrail prend en charge de nombreuses valeurs d'attribut pour filtrer votre recherche, notamment le nom de l'événement, le nom d'utilisateur et la source de l'événement.

Pour vous aider à rechercher des entrées de AWS KMS journal dans CloudTrail, AWS KMS remplit les champs d'entrée de CloudTrail journal suivants.

> **Note**
>
> À compter de décembre 2022, AWS KMS remplit les attributs Type de ressource et Nom de ressource dans toutes les opérations de gestion qui modifient une clé KMS particulière. Ces valeurs d'attribut peuvent être nulles dans les anciennes CloudTrail entrées pour les opérations suivantes : CreateAliasCreateGrantDeleteAliasDeleteImportedKeyMaterial, ImportKeyMaterial, ReplicateKey, RetireGrant, RevokeGrant, UpdateAlias, et UpdatePrimaryRegion.

Attribut	Valeur	Entrées de journal
Source de l'événement (`EventSource`)	`kms.amazonaws.com`	Toutes les opérations.
Type de ressource (`ResourceType`)	`AWS::KMS::Key`	Opérations de gestion qui modifient une clé KMS particulière, telles que `CreateKey` et `EnableKey`, mais pas `ListKeys`.
Nom de ressource (`ResourceName`)	ARN de clé (ou ID de clé et ARN de clé)	Opérations de gestion qui modifient une clé KMS

Attribut	Valeur	Entrées de journal
		particulière, telles que `CreateKey` et `EnableKey`, mais pas `ListKeys`.

Pour vous aider à trouver des entrées de journal pour les opérations de gestion sur des clés KMS spécifiques, AWS KMS enregistre l'ARN de la clé KMS affectée dans l'`responseElements.keyId`élément de l'entrée de journal, même si l'opération d' AWS KMS API ne renvoie pas l'ARN de la clé.

Par exemple, un appel réussi à l'DisableKeyopération ne renvoie aucune valeur dans la réponse, mais au lieu d'une valeur nulle, la `responseElements.keyId` valeur de l'entrée du DisableKey journal inclut l'ARN de la clé KMS désactivée.

Cette fonctionnalité a été ajoutée en décembre 2022 et concerne les entrées de CloudTrail journal suivantes : CreateAliasCreateGrantDeleteAlias, DeleteKey, DisableKey, EnableKey, EnableKeyRotation, ImportKeyMaterial, RotateKey, SynchronizeMultiRegionKey, TagResource, UntagResource, UpdateAlias, et UpdatePrimaryRegion.

Exclure AWS KMS des événements d'un parcours

Pour fournir un enregistrement de l'utilisation et de la gestion de leurs AWS KMS ressources, la plupart des AWS KMS utilisateurs s'appuient sur les événements d'un CloudTrail sentier. Le journal peut être une source de données précieuse pour l'audit d'événements critiques, tels que la création, la désactivation et la suppression AWS KMS keys, la modification de la politique en matière de clés et l'utilisation de vos clés KMS par les AWS services en votre nom. Dans certains cas, les métadonnées d'une entrée de CloudTrail journal, telles que le contexte de chiffrement d'une opération de chiffrement, peuvent vous aider à éviter ou à résoudre les erreurs.

Cependant, comme il AWS KMS peut générer un grand nombre d'événements, vous AWS CloudTrail permet d'exclure AWS KMS des événements d'un suivi. Ce paramètre par parcours exclut tous les AWS KMS événements ; vous ne pouvez pas exclure AWS KMS des événements particuliers.

> **Warning**
>
> L'exclusion d' AWS KMS événements d'un CloudTrail journal peut masquer les actions qui utilisent vos clés KMS. Soyez prudent lorsque vous accordez aux principaux l'autorisation `cloudtrail:PutEventSelectors` nécessaire pour effectuer cette opération.

Pour exclure AWS KMS des événements d'un parcours :

- Dans la CloudTrail console, utilisez le paramètre des événements du service Log Key Management lorsque vous créez un journal ou que vous le mettez à jour. Pour obtenir des instructions, reportez-vous à la section Logging Management Events AWS Management Console dans le guide de AWS CloudTrail l'utilisateur.
- Dans l' CloudTrail API, utilisez l'PutEventSelectorsopération. Ajoutez l'attribut `ExcludeManagementEventSources` à vos sélecteurs d'événements avec la valeur `kms.amazonaws.com`. Pour un exemple, voir Exemple : un journal qui n'enregistre pas les AWS Key Management Service événements dans le guide de AWS CloudTrail l'utilisateur.

Vous pouvez désactiver cette exclusion à tout moment en modifiant le paramètres de la console ou les sélecteurs d'événements d'un journal de suivi. Le sentier commencera alors à enregistrer AWS KMS les événements. Cependant, il ne peut pas récupérer les AWS KMS événements survenus pendant que l'exclusion était effective.

Lorsque vous excluez AWS KMS des événements à l'aide de la console ou de l'API, l'opération CloudTrail `PutEventSelectors` d'API qui en résulte est également enregistrée dans vos CloudTrail journaux. Si AWS KMS les événements n'apparaissent pas dans vos CloudTrail journaux, recherchez un `PutEventSelectors` événement dont l'`ExcludeManagementEventSources`attribut est défini sur`kms.amazonaws.com`.

Exemples d'entrées de AWS KMS journal

AWS KMS écrit des entrées dans votre CloudTrail journal lorsque vous appelez une AWS KMS opération et lorsqu'un AWS service appelle une opération en votre nom. AWS KMS écrit également une entrée lorsqu'il appelle une opération pour vous. Par exemple, il écrit une entrée lorsqu'il supprime une clé KMS dont vous avez programmé la suppression.

Les rubriques suivantes présentent des exemples d'entrées de CloudTrail journal pour les AWS KMS opérations.

Pour des exemples d'entrées de CloudTrail journal de demandes AWS KMS provenant de AWS Nitro Enclaves, consultez. Demandes de surveillance pour les enclaves Nitro

Rubriques

- CancelKeyDeletion
- ConnectCustomKeyStore
- CreateAlias
- CreateCustomKeyStore
- CreateGrant
- CreateKey
- Decrypt
- DeleteAlias
- DeleteCustomKeyStore
- DeleteExpiredKeyMaterial
- DeleteImportedKeyMaterial
- DeleteKey
- DescribeCustomKeyStores
- DescribeKey
- DisableKey
- DisableKeyRotation
- DisconnectCustomKeyStore
- EnableKey
- EnableKeyRotation
- Encrypt
- GenerateDataKey
- GenerateDataKeyPair
- GenerateDataKeyPairWithoutPlaintext
- GenerateDataKeyWithoutPlaintext
- GenerateMac

- GenerateRandom
- GetKeyPolicy
- GetKeyRotationStatus
- GetParametersForImport
- ImportKeyMaterial
- ListAliases
- ListGrants
- ListKeyRotations
- PutKeyPolicy
- ReEncrypt
- ReplicateKey
- RetireGrant
- RevokeGrant
- RotateKey
- RotateKeyOnDemand
- ScheduleKeyDeletion
- Sign (Signer)
- SynchronizeMultiRegionKey
- TagResource
- UntagResource
- UpdateAlias
- UpdateCustomKeyStore
- UpdateKeyDescription
- UpdatePrimaryRegion
- VerifyMac
- Vérification
- Premier EC2 exemple d'Amazon
- EC2 Exemple 2 d'Amazon

CancelKeyDeletion

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'CancelKeyDeletionopération. Pour plus d'informations sur la suppression AWS KMS keys, consultezSupprimer un AWS KMS key.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-07-27T21:53:17Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CancelKeyDeletion",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "e3452e68-d4b0-4ec7-a768-7ae96c23764f",
    "eventID": "d818bf03-6655-48e9-8b26-f279a07075fd",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

ConnectCustomKeyStore

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'`ConnectCustomKeyStore`opération. Pour plus d'informations sur la connexion d'un magasins de clés personnalisé, veuillez consulter Déconnecter un magasin de AWS CloudHSM clés.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-21T20:17:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ConnectCustomKeyStore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}
```

CreateAlias

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'CreateAliasopération. L'élément `resources` comprend des champs pour l'alias et les ressources clés KMS. Pour plus d'informations sur la création d'alias dans AWS KMS, consultezCréation d'alias.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-08-14T23:08:31Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateAlias",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "aliasName": "alias/ExampleAlias",
        "targetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "caec1e0c-ce03-419e-bdab-6ab1f7c57c01",
    "eventID": "2dd6e784-8286-46a6-befd-d64e5a02fb28",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
```

```
        },
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:alias/ExampleAlias"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

CreateCustomKeyStore

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'`CreateCustomKeyStore`opération sur un magasin de AWS CloudHSM clés. Pour plus d'informations sur les magasins de clés personnalisés, veuillez consulter Création d'un magasin de AWS CloudHSM clés.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-21T20:17:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateCustomKeyStore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "responseElements": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
```

```
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}
```

CreateGrant

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'CreateGrantopération. Pour plus d'informations sur la création de subventions dans AWS KMS, voirSubventions en AWS KMS.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
  "eventVersion": "1.02",
  "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn:aws:iam::111122223333:user/Alice",
      "accountId": "111122223333",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:53:12Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateGrant",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
      "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "constraints": {
          "encryptionContextSubset": {
              "ContextKey1": "Value1"
          }
      },
      "operations": ["Encrypt",
      "RetireGrant"],
```

```
      "granteePrincipal": "EX_PRINCIPAL_ID"
  },
  "responseElements": {
      "grantId": "f020fe75197b93991dc8491d6f19dd3cebb24ee62277a05914386724f3d48758",
      "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "f3c08808-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "5d529779-2d27-42b5-92da-91aaea1fc4b5",
  "readOnly": false,
  "resources": [{
      "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

CreateKey

Ces exemples montrent les entrées du AWS CloudTrail journal de l'CreateKeyopération.

Une entrée de `CreateKey` journal peut résulter d'une `CreateKey` demande ou de l'`CreateKey`opération d'une ReplicateKeydemande.

L'exemple suivant montre une entrée de CloudTrail journal pour une CreateKeyopération qui crée une clé KMS de chiffrement symétrique. Pour plus d'informations sur la création de clés KMS, veuillez consulter Création d'une clé KMS.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-08-10T22:38:27Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateKey",
```

```
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "description": "",
        "origin": "EXTERNAL",
        "bypassPolicyLockoutSafetyCheck": false,
        "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "keySpec": "SYMMETRIC_DEFAULT",
        "keyUsage": "ENCRYPT_DECRYPT"
    },
    "responseElements": {
        "keyMetadata": {
            "AWSAccountId": "111122223333",
            "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "creationDate": "Aug 10, 2022, 10:38:27 PM",
            "enabled": false,
            "description": "",
            "keyUsage": "ENCRYPT_DECRYPT",
            "keyState": "PendingImport",
            "origin": "EXTERNAL",
            "keyManager": "CUSTOMER",
            "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
            "keySpec": "SYMMETRIC_DEFAULT",
            "encryptionAlgorithms": [
                "SYMMETRIC_DEFAULT"
            ],
            "multiRegion": false
        }
    },
    "requestID": "1aef6713-0223-4ff7-9a6d-781360521930",
    "eventID": "36327b37-f4f6-40a9-92ab-48064ec905a2",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
```

```
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

L'exemple suivant montre le CloudTrail journal d'une `CreateKey` opération qui crée une clé KMS de chiffrement symétrique dans un magasin de AWS CloudHSM clés.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2021-10-14T17:39:50Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyUsage": "ENCRYPT_DECRYPT",
        "bypassPolicyLockoutSafetyCheck": false,
        "origin": "AWS_CLOUDHSM",
        "keySpec": "SYMMETRIC_DEFAULT",
        "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "customKeyStoreId": "cks-1234567890abcdef0",
        "description": ""
    },
    "responseElements": {
        "keyMetadata": {
            "aWSAccountId": "111122223333",
            "keyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
            "creationDate": "Oct 14, 2021, 5:39:50 PM",
            "enabled": true,
            "description": "",
            "keyUsage": "ENCRYPT_DECRYPT",
```

```
            "keyState": "Enabled",
            "origin": "AWS_CLOUDHSM",
            "customKeyStoreId": "cks-1234567890abcdef0",
            "cloudHsmClusterId": "cluster-1a23b4cdefg",
            "keyManager": "CUSTOMER",
            "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
            "keySpec": "SYMMETRIC_DEFAULT",
            "encryptionAlgorithms": [
                "SYMMETRIC_DEFAULT"
            ],
            "multiRegion": false
        }
    },
    "additionalEventData": {
        "backingKey": "{\"backingKeyId\":\"backing-key-id\"}"
    },
    "requestID": "4f0b185c-588c-4767-9e90-c618f7e13cad",
    "eventID": "c73964b8-703d-49e4-bd9e-f773d0ee1e65",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

L'exemple suivant montre le CloudTrail journal d'une `CreateKey` opération qui crée une clé KMS de chiffrement symétrique dans un magasin de clés externe.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
```

```
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2022-09-07T22:37:45Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "tags": [],
        "keyUsage": "ENCRYPT_DECRYPT",
        "description": "",
        "origin": "EXTERNAL_KEY_STORE",
        "multiRegion": false,
        "keySpec": "SYMMETRIC_DEFAULT",
        "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "bypassPolicyLockoutSafetyCheck": false,
        "customKeyStoreId": "cks-1234567890abcdef0",
        "xksKeyId": "bb8562717f809024"
    },
    "responseElements": {
        "keyMetadata": {
            "aWSAccountId": "111122223333",
            "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "creationDate": "Dec 7, 2022, 10:37:45 PM",
            "enabled": true,
            "description": "",
            "keyUsage": "ENCRYPT_DECRYPT",
            "keyState": "Enabled",
            "origin": "EXTERNAL_KEY_STORE",
            "customKeyStoreId": "cks-1234567890abcdef0",
            "keyManager": "CUSTOMER",
            "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
            "keySpec": "SYMMETRIC_DEFAULT",
            "encryptionAlgorithms": [
                "SYMMETRIC_DEFAULT"
            ],
            "multiRegion": false,
            "xksKeyConfiguration": {
                "id": "bb8562717f809024"
            }
```

```
        }
    },
    "requestID": "ba197c82-3ac7-487a-8ff4-7736bbeb1316",
    "eventID": "838ad5f4-5fdd-4044-afd7-4dbd88c6af56",
    "readOnly": false,
    "resources": [
        {
            "accountId": "227179770375",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-east-1:227179770375:key/39c5eb22-
f37c-4956-92ca-89e8f8b57ab2"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Ces exemples montrent les entrées du AWS CloudTrail journal pour l'opération de déchiffrement.

L'entrée du CloudTrail journal d'une `Decrypt` opération inclut toujours le `encryptionAlgorithm` in `requestParameters` même si l'algorithme de chiffrement n'a pas été spécifié dans la demande. Le texte chiffré de la demande et le texte brut de la réponse sont omis.

Rubriques

- Déchiffrer avec une clé de chiffrement symétrique standard
- Échec lors du déchiffrement avec une clé de chiffrement symétrique standard
- Déchiffrer avec une clé KMS dans un magasin de AWS CloudHSM clés
- Déchiffrer avec une clé KMS dans un magasin de clés externe
- Échec lors du déchiffrement avec une clé KMS dans un magasin de clés externe

Déchiffrer avec une clé de chiffrement symétrique standard

Voici un exemple d'entrée de CloudTrail journal pour une `Decrypt` opération utilisant une clé de chiffrement symétrique standard.

```
{
```

```
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:45:00Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "encryptionContext": {
            "Department": "Engineering",
            "Project": "Alpha"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "12345126-30d5-4b28-98b9-9153da559963",
    "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
```

```
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
```

Échec lors du déchiffrement avec une clé de chiffrement symétrique standard

L'exemple d'entrée de CloudTrail journal suivant enregistre l'échec d'une `Decrypt` opération avec une clé KMS de chiffrement symétrique standard. L'exception (`errorCode`) et le message d'erreur (`errorMessage`) sont inclus pour vous aider à résoudre l'erreur.

Dans ce cas, la clé KMS de chiffrement symétrique spécifiée dans la requête `Decrypt` n'était pas la clé KMS de chiffrement symétrique utilisée pour chiffrer les données.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-11-24T18:57:43Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "errorCode": "IncorrectKeyException"
    "errorMessage": "The key ID in the request does not identify a CMK that can perform
 this operation.",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "encryptionContext": {
            "Department": "Engineering",
            "Project": "Alpha"
        }
```

```
    },
    "responseElements": null,
    "requestID": "22345126-30d5-4b28-98b9-9153da559963",
    "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Déchiffrer avec une clé KMS dans un magasin de AWS CloudHSM clés

L'exemple d'entrée de CloudTrail journal suivant enregistre une `Decrypt` opération avec une clé KMS dans un magasin de AWS CloudHSM clés. Toutes les entrées de journal relatives aux opérations cryptographiques effectuées avec une clé KMS dans un magasin de clés personnalisé incluent un `additionalEventData` champ avec le `customKeyStoreId` et`backingKeyId`. La valeur renvoyée dans le `backingKeyId` champ est l'attribut clé `id` CloudHSM. `additionalEventData` n'est pas spécifié dans la requête.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-26T23:41:27Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
```

```
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "encryptionContext": {
            "Department": "Development",
            "Purpose": "Test"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "requestID": "e1b881f8-2048-41f8-b6cc-382b7857ec61",
    "eventID": "a79603d5-4cde-46fc-819c-a7cf547b9df4",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Déchiffrer avec une clé KMS dans un magasin de clés externe

L'exemple d'entrée de CloudTrail journal suivant enregistre une `Decrypt` opération avec une clé KMS dans un magasin de clés externe. Outre `customKeyStoreId`, le champ `additionalEventData` inclut l'ID de clé externe (`XksKeyId`). `additionalEventData` n'est pas spécifié dans la requête.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
```

```
        "userName": "Alice"
    },
    "eventTime": "2022-11-24T00:26:58Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "encryptionContext": {
            "Department": "Engineering",
            "Purpose": "Test"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreId": "cks-9876543210fedcba9",
        "xksKeyId": "abc01234567890fe"
    },
    "requestID": "f1b881f8-2048-41f8-b6cc-382b7857ec61",
    "eventID": "b79603d5-4cde-46fc-819c-a7cf547b9df4",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Échec lors du déchiffrement avec une clé KMS dans un magasin de clés externe

L'exemple d'entrée de CloudTrail journal suivant enregistre l'échec d'une demande d'`Decrypt`opération avec une clé KMS dans un magasin de clés externe. CloudWatch enregistre les demandes qui échouent, en plus des demandes réussies. Lors de l'enregistrement d'un échec,

l'entrée du CloudTrail journal inclut l'exception (ErrorCode) et le message d'erreur qui l'accompagne (ErrorMessage).

Si la requête échouée a atteint votre proxy de magasin de clés externe, comme dans cet exemple, vous pouvez utiliser la valeur `requestId` pour associer la requête échouée à une requête correspondante des journaux de votre proxy de magasin de clés externe, si votre proxy les fournit.

Pour obtenir de l'aide concernant les requêtes `Decrypt` provenant de magasins de clés externes, veuillez consulter la rubrique Erreurs de déchiffrement.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-11-24T00:26:58Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "errorCode": "KMSInvalidStateException",
    "errorMessage": "The external key store proxy rejected the request because the
 specified ciphertext or additional authenticated data is corrupted, missing, or
 otherwise invalid.",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "encryptionContext": {
            "Department": "Engineering",
            "Purpose": "Test"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreId": "cks-9876543210fedcba9",
        "xksKeyId": "abc01234567890fe"
```

```
    },
    "requestID": "f1b881f8-2048-41f8-b6cc-382b7857ec61",
    "eventID": "b79603d5-4cde-46fc-819c-a7cf547b9df4",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

DeleteAlias

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'DeleteAliasopération. Pour plus d'informations sur la suppression d'alias, veuillez consulter Supprimer un alias.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2014-11-04T00:52:27Z"
            }
        }
    },
```

```
    "eventTime": "2014-11-04T00:52:27Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteAlias",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "aliasName": "alias/my_alias"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "d9542792-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "12f48554-bb04-4991-9cfc-e7e85f68eda0",
    "readOnly": false,
    "resources": [{
        "ARN": "arn:aws:kms:us-east-1:111122223333:alias/my_alias",
        "accountId": "111122223333"
    },
    {
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


DeleteCustomKeyStore

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'`DeleteCustomKeyStore`opération. Pour plus d'informations sur les magasins de clés personnalisés, veuillez consulter Supprimer un magasin AWS CloudHSM de clés.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
```

```
        "userName": "Alice"
    },
    "eventTime": "2021-10-21T20:17:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteCustomKeyStore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}
```


DeleteExpiredKeyMaterial

Lorsque vous importez du matériel clé dans une AWS KMS key (clé KMS), vous pouvez définir une date et une heure d'expiration pour ce matériel clé. AWS KMS enregistre une entrée dans votre CloudTrail journal lorsque vous importez le matériel clé (avec les paramètres d'expiration) et lorsque vous AWS KMS supprimez le matériel clé expiré. Pour plus d'informations sur la création de clés KMS avec des éléments de clé importés, veuillez consulter Importation de matériel clé pour les AWS KMS clés.

L'exemple suivant montre une entrée de AWS CloudTrail journal générée lors de la AWS KMS suppression du contenu clé expiré.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
```

```
    "eventTime": "2025-05-22T19:55:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteExpiredKeyMaterial",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "cfa932fd-0d3a-4a76-a8b8-616863a2b547",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "eventCategory": "Management"
}
```


DeleteImportedKeyMaterial

Si vous importez du matériel clé dans une clé KMS, vous pouvez supprimer le matériel clé importé à tout moment en utilisant cette DeleteImportedKeyMaterialopération. Lorsque vous supprimez des éléments de clé importés d'une clé KMS, l'état de la clé KMS passe à `PendingImport` et la clé KMS ne peut être utilisée dans aucune opération cryptographique. Pour en savoir plus, consultez Supprimer le matériel clé importé.

L'exemple suivant montre une entrée de AWS CloudTrail journal générée pour l'`DeleteImportedKeyMaterial`opération.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
```

```
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:45:08Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteImportedKeyMaterial",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "dcf0e82f-dad0-4622-a378-a5b964ad42c1",
    "eventID": "2afbb991-c668-4641-8a00-67d62e1fecbd",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
```

```
}
```

DeleteKey

Ces exemples montrent l'entrée de AWS CloudTrail journal qui est générée lorsqu'une clé KMS est supprimée. Pour supprimer une clé KMS, vous devez utiliser l'ScheduleKeyDeletionopération. Une fois le délai d'attente spécifié expiré, AWS KMS supprime la clé KMS et enregistre une entrée comme celle-ci dans votre CloudTrail journal pour enregistrer cet événement.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

Pour un exemple de l'entrée du CloudTrail journal de l'`ScheduleKeyDeletion`opération, consultezScheduleKeyDeletion. Pour plus d'informations sur la suppression de clés KMS, veuillez consulter Supprimer un AWS KMS key.

L'exemple d'entrée de CloudTrail journal suivant enregistre une `DeleteKey` opération sur une clé KMS contenant du contenu clé AWS KMS.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2020-07-31T00:07:00Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "b25f9cda-74e1-4458-847b-4972a0bf9668",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
```

```
        }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
     "managementEvent": true,
    "eventCategory": "Management"
}
```

L'entrée de CloudTrail journal suivante enregistre `DeleteKey` l'opération d'une clé KMS dans un magasin de clés AWS CloudHSM personnalisé.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-10-26T23:41:27Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "additionalEventData": {
        "customKeyStoreId": "cks-1234567890abcdef0",
        "clusterId": "cluster-1a23b4cdefg",
        "backingKeys": "[{\"backingKeyId\":\"backing-key-id\"}]",
        "backingKeysDeletionStatus": "[{\"backingKeyId\":\"backing-key-id\",
\"deletionStatus\":\"SUCCESS\"}]"
    },
    "eventID": "1234585c-4b0c-4340-ab11-662414b79239",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
```

```
        }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
    "managementEvent": true,
    "eventCategory": "Management"
}
```

DescribeCustomKeyStores

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'`DescribeCustomKeyStores`opération. Pour plus d'informations sur les magasins de clés personnalisés, veuillez consulter Afficher un magasin AWS CloudHSM de clés.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-21T20:17:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeCustomKeyStores",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "responseElements": null,
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "2ea1735f-628d-43e3-b2ee-486d02913a78",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}
```

DescribeKey

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'DescribeKeyopération. AWS KMS enregistre une entrée similaire à la suivante lorsque vous appelez l'`DescribeKey`opération ou que vous affichez les clés KMS dans la AWS KMS console. Cet appel est le résultat de l'affichage d'une clé dans la console AWS KMS de gestion.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-09-26T18:01:36Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "requestID": "12345126-30d5-4b28-98b9-9153da559963",
    "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

DisableKey

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'DisableKeyopération. Pour plus d'informations sur l'activation et la désactivation AWS KMS keys dans AWS KMS, consultezActiver et désactiver les touches.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.


```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:52:43Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DisableKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "12345126-30d5-4b28-98b9-9153da559963",
    "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": false,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
```

```
}
```

DisableKeyRotation

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'DisableKeyRotationopération. Pour plus d'informations sur la rotation automatique des clés, consultez Rotation AWS KMS keys.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-09-01T19:31:39Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DisableKeyRotation",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "requestID": "d6a9351a-ed6e-4581-88d1-2a9a8a538497",
    "eventID": "6313164c-83aa-4cc3-9e1a-b7c426f7a5b1",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
```

```
    "eventCategory": "Management"
}
```

DisconnectCustomKeyStore

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'`DisconnectCustomKeyStore`opération. Pour plus d'informations sur les magasins de clés personnalisés, veuillez consulter Déconnecter un magasin de AWS CloudHSM clés.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-21T20:17:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DisconnectCustomKeyStore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}
```

EnableKey

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'EnableKeyopération. Pour plus d'informations sur l'activation et la désactivation AWS KMS keys dans AWS KMS, voirActiver et désactiver les touches.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.


```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:52:20Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "EnableKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "d528a6fb-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "be393928-3629-4370-9634-567f9274d52e",
    "readOnly": false,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
```

```
}
```

EnableKeyRotation

L'exemple suivant montre une entrée dans le AWS CloudTrail journal d'un appel à l'EnableKeyRotationopération. Pour un exemple d'entrée de CloudTrail journal écrite lors de la rotation de la clé, consultezRotateKey. Pour plus d'informations sur la rotation de AWS KMS keys, veuillez consulter Rotation AWS KMS keys.

> **Note**
>
> rotation-periodIl s'agit d'un paramètre de demande facultatif. Si vous ne spécifiez pas de période de rotation lorsque vous activez la rotation automatique des clés, la valeur par défaut est de 365 jours.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-07-25T23:41:56Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "EnableKeyRotation",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "rotationPeriodInDays": 180
    },
    "responseElements": {
```

```
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "81f5b794-452b-4d6a-932b-68c188165273",
    "eventID": "fefc43a7-8e06-419f-bcab-b3bf18d6a401",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Encrypt

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'opération Encrypt.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:46:16Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Encrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "encryptionContext": {
            "Department": "Engineering"
        },
```

```
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "f3423043-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "91235988-eb87-476a-ac2c-0cdc244e6dca",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
}
```


GenerateDataKey

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GenerateDataKeyopération.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:46:16Z",
```

```
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "keySpec": "AES_256",
        "encryptionContext": {
            "Department": "Engineering",
            "Project": "Alpha"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
}
```


GenerateDataKeyPair

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GenerateDataKeyPairopération. Cet exemple enregistre une opération qui génère une paire de clés RSA chiffrée sous une AWS KMS key de chiffrement symétrique.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:46:16Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyPair",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyPairSpec": "RSA_3072",
        "encryptionContext": {
            "Project": "Alpha"
        },
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
    "eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
```

```
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
```

GenerateDataKeyPairWithoutPlaintext

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GenerateDataKeyPairWithoutPlaintextopération. Cet exemple enregistre une opération qui génère une paire de clés RSA qui est chiffrée sous une AWS KMS key de chiffrement symétrique.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:46:16Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyPairWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyPairSpec": "RSA_4096",
        "encryptionContext": {
            "Index": "5"
        },
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
    "eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
```

```
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
```

GenerateDataKeyWithoutPlaintext

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GenerateDataKeyWithoutPlaintextopération.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:46:16Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
        "keySpec": "AES_256",
        "encryptionContext": {
            "Project": "Alpha"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "d6b8e411-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "f7734272-9ec5-4c80-9f36-528ebbe35e4a",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
}
```

GenerateMac

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GenerateMacopération.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
     },
```

```
    "eventTime": "2022-12-23T19:26:54Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateMac",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "macAlgorithm": "HMAC_SHA_512",
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

GenerateRandom

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GenerateRandomopération. Comme cette opération n'utilise pas un AWS KMS key, le `resources` champ est vide.

```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:52:37Z",
    "eventSource": "kms.amazonaws.com",
```

```
    "eventName": "GenerateRandom",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "df1e3de6-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "239cb9f7-ae05-4c94-9221-6ea30eef0442",
    "readOnly": true,
    "resources": [],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

GetKeyPolicy

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GetKeyPolicyopération. Pour plus d'informations sur l'affichage de la politique de clé pour une clé KMS, veuillez consulter Afficher une politique clé.

```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:50:30Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GetKeyPolicy",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "policyName": "default"
    },
    "responseElements": null,
    "requestID": "93746dd6-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "4aa7e4d5-d047-452a-a5a6-2cce282a7e82",
```

```
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

GetKeyRotationStatus

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'GetKeyRotationStatusopération. Pour plus d'informations sur la rotation automatique et à la demande des éléments clés d'une clé KMS, consultezRotation AWS KMS keys.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2024-02-20T19:16:45Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GetKeyRotationStatus",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "requestID": "12f9b7e8-49b9-4c1c-a7e3-34ac0cdf0467",
    "eventID": "3d082126-9e7d-4167-8372-a6cfcbed4be6",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
}
```

GetParametersForImport

L'exemple suivant montre une entrée de AWS CloudTrail journal générée lorsque vous utilisez l'GetParametersForImportopération. Cette opération renvoie la clé publique et le jeton d'importation que vous utilisez lorsque vous importez des éléments de clé dans une clé KMS. La même CloudTrail entrée est enregistrée lorsque vous utilisez l'`GetParametersForImport`opération ou que vous utilisez la AWS KMS console pour télécharger la clé publique et le jeton d'importation.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2020-07-25T23:58:23Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GetParametersForImport",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "wrappingAlgorithm": "RSAES_OAEP_SHA_256",
        "wrappingKeySpec": "RSA_2048"
```

```
    },
    "responseElements": null,
    "requestID": "b5786406-e3c7-43d6-8d3c-6d5ef96e2278",
    "eventID": "4023e622-0c3e-4324-bdef-7f58193bba87",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

ImportKeyMaterial

L'exemple suivant montre une entrée de AWS CloudTrail journal générée lorsque vous utilisez l'ImportKeyMaterialopération. La même CloudTrail entrée est enregistrée lorsque vous utilisez l'`ImportKeyMaterial`opération ou que vous utilisez la AWS KMS console pour importer du matériel clé dans un AWS KMS key.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2025-05-21T05:42:31Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ImportKeyMaterial",
    "awsRegion": "us-west-2",
```

```
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "validTo": "May 21, 2025, 5:47:45 AM",
        "expirationModel": "KEY_MATERIAL_EXPIRES",
        "importType": "NEW_KEY_MATERIAL",
        "keyMaterialDescription": "ExampleKeyMaterialA"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "keyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "89e10ee7-a612-414d-95a2-a128346969fd",
    "eventID": "c7abd205-a5a2-4430-bbfa-fc10f3e2d79f",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
```


ListAliases

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'ListAliasesopération. Comme cette opération n'utilise aucun alias en particulier ou AWS KMS key que le `resources` champ est vide. Pour plus d'informations sur l'affichage des alias dans AWS KMS, consultezTrouvez le nom d'alias et l'ARN de l'alias pour une clé KMS.

```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:51:45Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ListAliases",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "limit": 5,
        "marker":
 "eyJiIjoiYWxpYXMvZTU0Y2MxOTMtYTMwNC00YzEwLTliZWItYTJjZjA3NjA2OTJhIiwiYSI6ImFsaWFzL2U1NGNjMTkzL
    },
    "responseElements": null,
    "requestID": "bfe6c190-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "a27dda7b-76f1-4ac3-8b40-42dfba77bcd6",
    "readOnly": true,
    "resources": [],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


ListGrants

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'ListGrantopération. Pour plus d'informations sur les subventions accordées AWS KMS, voirSubventions en AWS KMS.


```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
```

```
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:52:49Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ListGrants",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "marker":
 "eyJncmFudElkIjoiMWY4M2U2ZmM0YTY2NDgxYjQ2Yzc4MTdhM2Y4YmQwMDFkZDNiYmQ1MGVlYTMyY2RmOWFiNWY1Nzc1N
\u003d\u003d",
        "limit": 10
    },
    "responseElements": null,
    "requestID": "e5c23960-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "d24380f5-1b20-4253-8e92-dd0492b3bd3d",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


ListKeyRotations

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'ListKeyRotationsopération. Pour plus d'informations sur la rotation automatique et à la demande des éléments clés d'une clé KMS, consultezRotation AWS KMS keys.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
```

```
    },
    "eventTime": "2025-05-21T05:42:35Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ListKeyRotations",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "includeKeyMaterial": "ALL_KEY_MATERIAL"
    },
    "responseElements": null,
    "requestID": "99c88d32-f2db-455e-8a9a-23855258a452",
    "eventID": "8ce0e74b-b9c7-45a2-96ef-83136d38068e",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
}
```


PutKeyPolicy

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'PutKeyPolicyopération. Pour plus d'informations sur la mise à jour d'une stratégie de clé, consultez Modifier une politique clé.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
```

```
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-09-01T20:06:16Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "PutKeyPolicy",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "policyName": "default",
        "policy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Id\" : \"key-default-1\",\n
 \"Statement\" : [ {\n    \"Sid\" : \"Enable IAM User Permissions\",\n    \"Effect\" :
 \"Allow\",\n    \"Principal\" : {\n      \"AWS\" : \"arn:aws:iam::111122223333:root
\"\n    },\n    \"Action\" : \"kms:*\",\n    \"Resource\" : \"*\"\n  } ]\n}",
        "bypassPolicyLockoutSafetyCheck": false
    },
    "responseElements": null,
    "requestID": "7bb906fa-dc21-4350-b65c-808ff0f72f55",
    "eventID": "c217db1f-903f-4a2f-8f88-9580182d6313",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

ReEncrypt

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'ReEncryptopération. Le `resources` champ de cette entrée de journal en spécifie deux AWS KMS keys, la clé KMS source et la clé KMS de destination, dans cet ordre.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-22T19:34:55Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ReEncrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "sourceEncryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "sourceEncryptionContext": {
            "Project": "Alpha",
            "Department": "Engineering"
        },
        "destinationKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "destinationEncryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "destinationEncryptionContext": {
            "Level": "3A"
        }
    },
    "responseElements": null,
    "additionalEventData": {
        "destinationKeyMaterialId":
 "96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068",
        "sourceKeyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "03769fd4-acf9-4b33-adf3-2ab8ca73aadf",
    "eventID": "542d9e04-0e8d-4e05-bf4b-4bdeb032e6ec",
```

```
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
```

ReplicateKey

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'ReplicateKeyopération. Une `ReplicateKey` demande entraîne une `ReplicateKey` opération et une CreateKeyopération.

Pour plus d'informations sur la réplication de clés multi-région, veuillez consulter Création de répliques de clés multirégionales.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
```

```
        "userName": "Alice"
    },
    "eventTime": "2020-11-18T01:29:18Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ReplicateKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "replicaRegion": "us-west-2",
        "bypassPolicyLockoutSafetyCheck": false,
        "description": ""
    },
    "responseElements": {
        "replicaKeyMetadata": {
            "aWSAccountId": "111122223333",
            "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "creationDate": "Nov 18, 2020, 1:29:18 AM",
            "enabled": false,
            "description": "",
            "keyUsage": "ENCRYPT_DECRYPT",
            "keyState": "Creating",
            "origin": "AWS_KMS",
            "keyManager": "CUSTOMER",
            "keySpec": "SYMMETRIC_DEFAULT",
            "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
            "encryptionAlgorithms": [
                "SYMMETRIC_DEFAULT"
            ],
            "multiRegion": true,
            "multiRegionConfiguration": {
                "multiRegionKeyType": "REPLICA",
                "primaryKey": {
                    "arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                    "region": "us-east-1"
                },
                "replicaKeys": [
                    {
                        "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
                        "region": "us-west-2"
                    }
                ]
            }
        },
        "replicaPolicy": "{\n  \"Version\":\"2012-10-17\",\n  \"Statement\":[{\n
 \"Effect\":\"Allow\",\n    \"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:user/
Alice\"},\n    \"Action\":\"kms:*\",\n    \"Resource\":\"*\"\n  }, {\n    \"Effect
\":\"Allow\",\n    \"Principal\":{\"AWS\":\"arn:aws:iam::012345678901:user/Bob\"},\n
    \"Action\":\"kms:CreateGrant\",\n    \"Resource\":\"*\"\n  }, {\n    \"Effect\":
\"Allow\",\n    \"Principal\":{\"AWS\":\"arn:aws:iam::012345678901:user/Charlie\"},\n
  \"Action\":\"kms:Encrypt\",\n    \"Resource\":\"*\"\n}]\n}",
    },
    "requestID": "abcdef68-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "fedcba44-6773-4f96-8763-1993aec9ae6a",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


RetireGrant

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'RetireGrantopération. Pour plus d'informations concernant le retrait d'octrois, veuillez consulter Retrait et révocation d'octrois.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
```

```
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-09-01T19:39:33Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RetireGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData": {
        "grantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a"
    },
    "requestID": "1d274d57-5697-462c-a004-f25fcc29fa26",
    "eventID": "0771bcfb-3e24-4332-9ac8-e1c06563eecf",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

RevokeGrant

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'RevokeGrantopération. Pour plus d'informations sur la révocation d'octrois, veuillez consulter Retrait et révocation d'octrois.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
```

```
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-09-01T19:35:17Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RevokeGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "grantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a"
    },
    "responseElements": null,
    "requestID": "59d94c03-c5b7-428d-ae6e-f2c4b47d2917",
    "eventID": "07a23a39-6526-4ae2-b31e-d35fbe9e24ee",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


RotateKey

Ces exemples montrent les entrées du AWS CloudTrail journal pour les opérations qui font l'objet d'une rotation AWS KMS keys. Pour plus d'informations sur la rotation des clés KMS, veuillez consulter Rotation AWS KMS keys.

L'exemple suivant montre une entrée de CloudTrail journal pour l'opération qui fait pivoter une clé KMS de chiffrement symétrique sur laquelle la rotation automatique des clés est activée. Pour plus d'informations sur l'activation de la rotation automatique, consultezRotation AWS KMS keys.

Pour un exemple de l'entrée du CloudTrail journal qui enregistre l'`EnableKeyRotation`opération, consultez[EnableKeyRotation](#).


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2025-05-20T20:44:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RotateKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "a24b3967-ddad-417f-9b22-2332b918db06",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "rotationType": "AUTOMATIC",
        "keyOrigin": "AWS_KMS",
        "previousKeyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0",
        "currentKeyMaterialId":
 "96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068",
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "eventCategory": "Management"
}
```

L'exemple suivant montre une entrée de CloudTrail journal pour une rotation à la demande initiée par l'RotateKeyOnDemandopération. Pour plus d'informations sur la rotation des clés KMS de chiffrement symétrique à la demande, consultezEffectuez une rotation des touches à la demande.

Pour un exemple de l'entrée du CloudTrail journal qui enregistre l'`RotateKeyOnDemand`opération, consultezRotateKeyOnDemand.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2025-05-20T20:44:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RotateKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "a24b3967-ddad-417f-9b22-2332b918db06",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "rotationType": "ON_DEMAND",
        "keyOrigin": "EXTERNAL",
        "previousKeyMaterialId":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0",
        "currentKeyMaterialId":
 "96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068",
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
```

```
    "eventCategory": "Management"
}
```

RotateKeyOnDemand

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'RotateKeyOnDemandopération. Pour un exemple d'entrée de CloudTrail journal écrite lors de la rotation de la clé, consultezRotateKey. Pour plus d'informations sur la rotation à la demande du contenu clé d'une clé KMS, consultezEffectuez une rotation des touches à la demande.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2024-02-20T17:41:57Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RotateKeyOnDemand",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "9e1dee86-eb84-42fd-8f25-e3fc7dbb32c8",
    "eventID": "00a09fbc-20d6-4a58-9b92-7da85984ab77",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
```

```
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
}
```

ScheduleKeyDeletion

Ces exemples montrent les entrées du AWS CloudTrail journal de l'ScheduleKeyDeletionopération.

Pour un exemple d'entrée de CloudTrail journal écrite lorsque la clé est supprimée, consultezDeleteKey. Pour plus d'informations sur la suppression de AWS KMS keys, veuillez consulter Supprimer un AWS KMS key.

L'exemple suivant enregistre une demande `ScheduleKeyDeletion` de clé KMS à région unique.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2021-03-23T18:58:30Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ScheduleKeyDeletion",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "pendingWindowInDays": 20,
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
```

```
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "keyState": "PendingDeletion",
        "deletionDate": "Apr 12, 2021 18:58:30 PM"
    },
    "requestID": "ee408f36-ea01-422b-ac14-b0f147c68334",
    "eventID": "3c4226b0-1e81-48a8-a333-7fa5f3cbd118",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

L'exemple suivant enregistre une demande `ScheduleKeyDeletion` de clé KMS multi-région avec des clés de réplica.

Parce qu'une clé multirégionale AWS KMS ne sera pas supprimée tant que toutes ses répliques ne seront pas supprimées, dans le `responseElements` champ, la valeur `keyState` est `PendingReplicaDeletion` et le `deletionDate` champ est omis.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2021-10-28T17:59:05Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ScheduleKeyDeletion",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
```

```
    "requestParameters": {
        "pendingWindowInDays": 30,
        "keyId": "mrk-1234abcd12ab34cd56ef1234567890ab"
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "keyState": "PendingReplicaDeletion",
        "pendingWindowInDays": 30
    },
    "requestID": "12341411-d846-42a6-a476-b1cbe3011f89",
    "eventID": "abcda5f-396d-494c-9380-0c47860df5f1",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


L'exemple suivant enregistre une `ScheduleKeyDeletion` demande de clé KMS dans un magasin de clés AWS CloudHSM personnalisé.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2021-10-26T23:25:25Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ScheduleKeyDeletion",
```

```
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "pendingWindowInDays": 30
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321",
        "deletionDate": "Nov 2, 2021, 11:25:25 PM",
        "keyState": "PendingDeletion",
        "pendingWindowInDays": 30
    },
    "additionalEventData": {
        "customKeyStoreId": "cks-1234567890abcdef0",
        "clusterId": "cluster-1a23b4cdefg",
        "backingKeys": "[{\"backingKeyId\":\"backing-key-id\"}]"
    },
    "requestID": "abcd9f60-2c9c-4a0b-a456-d5d998f7f321",
    "eventID": "ca01996a-01b0-4edd-bbbb-25d7b6d1a6fa",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


Sign (Signer)

Ces exemples montrent les entrées du AWS CloudTrail journal pour l'opération de signature.

L'exemple suivant montre une entrée de CloudTrail journal pour une opération de signature qui utilise une clé RSA KMS asymétrique pour générer une signature numérique pour un fichier.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-03-07T22:36:44Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Sign",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "messageType": "RAW",
        "keyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "signingAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
    },
    "responseElements": null,
    "requestID": "8d0b35e0-46cf-48b9-be99-bf2ebc9ab9fb",
    "eventID": "107b3cac-b125-4556-9702-12a2b9afcff7",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


SynchronizeMultiRegionKey

L'exemple suivant montre une entrée de AWS CloudTrail journal générée lors de AWS KMS la synchronisation d'une clé multirégionale. La synchronisation implique des appels interrégionaux

pour copier les propriétés partagées d'une clé primaire multirégionale dans ses clés de réplique. AWS KMS synchronise périodiquement les clés multirégionales pour s'assurer que toutes les clés multirégionales associées ont le même contenu clé.

L'`resources`élément de l'entrée du CloudTrail journal inclut l'ARN de la clé primaire multirégionale, y compris son Région AWS. Les clés de réplica multi-région associées et leurs régions ne sont pas répertoriées dans cette entrée de journal.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2020-11-18T02:04:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "SynchronizeMultiRegionKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "12345681-de97-42e9-bed0-b02ae1abd8dc",
    "eventID": "abcdec99-2b5c-4670-9521-ddb8f031e146",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
```

```
    "eventCategory": "Management"
}
```

TagResource

L'exemple suivant montre une entrée de AWS CloudTrail journal d'un appel à l'TagResourceopération visant à ajouter une balise avec une clé de balise `Department` et une valeur de balise de`IT`.

Pour un exemple d'entrée de `UntagResource` CloudTrail journal écrite lors de la rotation de la clé, consultezUntagResource. Pour plus d'informations sur le balisage AWS KMS keys, consultezTags dans AWS KMS.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-07-01T21:19:25Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "TagResource",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "tags": [
            {
                "tagKey": "Department",
                "tagValue": "IT"
            }
        ]
    },
    "responseElements": null,
    "requestID": "b942584a-f77d-4787-9feb-b9c5be6e746d",
    "eventID": "0a091b9b-0df5-4cf9-b667-6f2879532b8f",
    "readOnly": false,
```

```
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

UntagResource

L'exemple suivant montre une entrée de AWS CloudTrail journal d'un appel à l'UntagResourceopération de suppression d'une balise dont la clé de balise est égale à`Dept`.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

Pour un exemple d'entrée de `TagResource` CloudTrail journal, voirTagResource. Pour plus d'informations sur l'étiquetage AWS KMS keys, veuillez consulter Tags dans AWS KMS.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-07-01T21:19:19Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UntagResource",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
        "tagKeys": [
            "Dept"
        ]
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "cb1d507b-6015-47f4-812b-179713af8068",
    "eventID": "0b00f4b0-036e-411d-aa75-87eb4a35a4b3",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

UpdateAlias

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'UpdateAliasopération. L'élément `resources` comprend des champs pour l'alias et les ressources clés KMS. Pour plus d'informations sur la création d'alias dans AWS KMS, consultezCréation d'alias.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
         "principalId": "EX_PRINCIPAL_ID",
         "arn": "arn:aws:iam::111122223333:user/Alice",
         "accountId": "111122223333",
         "accessKeyId": "EXAMPLE_KEY_ID",
         "userName": "Alice"
    },
```

```
    "eventTime": "2020-11-13T23:18:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdateAlias",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "aliasName": "alias/my_alias",
        "targetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "d9472f40-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "f72d3993-864f-48d6-8f16-e26e1ae8dff0",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:alias/my_alias"
        },
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


UpdateCustomKeyStore

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'`UpdateCustomKeyStore`opération de mise à jour de l'ID de cluster pour un magasin de clés personnalisé. Pour plus d'informations sur les magasins de clés personnalisés, veuillez consulter Modifier les paramètres du magasin AWS CloudHSM clé.


```
{
    "eventVersion": "1.08",
```

```
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-21T20:17:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdateCustomKeyStore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreId": "cks-1234567890abcdef0",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}
```


UpdateKeyDescription

L'exemple suivant montre une entrée de AWS CloudTrail journal générée en appelant l'UpdateKeyDescriptionopération.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
```

```
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-09-01T19:22:40Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdateKeyDescription",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "description": "New key description"
    },
    "responseElements": null,
    "requestID": "8c3c1f8b-336d-4896-b034-4eb9916bc9b3",
    "eventID": "f5f3d548-2e9e-4658-8427-9dcb5b1ea791",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


UpdatePrimaryRegion

L'exemple suivant montre les entrées de AWS CloudTrail journal générées en appelant l'UpdatePrimaryRegionopération sur une clé multirégionale.

L'`UpdatePrimaryRegion`opération écrit deux entrées de CloudTrail journal : l'une dans la région avec la clé primaire multirégionale qui est convertie en clé de réplique, et l'autre dans la région avec une clé de réplique multirégion convertie en clé primaire.

CloudTrail les entrées du journal pour cette opération enregistrées en décembre 2022 ou après cette date incluent l'ARN de la clé KMS affectée dans la `responseElements.keyId` valeur, même si cette opération ne renvoie pas l'ARN de la clé.

L'exemple suivant montre une entrée de CloudTrail journal pour la `UpdatePrimaryRegion` région où la clé multirégionale est passée d'une clé primaire à une clé de réplique (us-west-2). Le champ `primaryRegion` montre la région qui héberge désormais la clé primaire (ap-northeast-1).


```
{
    "eventVersion": "1.08",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
    },
    "eventTime": "2021-03-10T20:23:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdatePrimaryRegion",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "primaryRegion": "ap-northeast-1"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "ee408f36-ea01-422b-ac14-b0f147c68334",
    "eventID": "3c4226b0-1e81-48a8-a333-7fa5f3cbd118",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333"
```

```
}
```

L'exemple suivant représente l'entrée de CloudTrail journal pour `UpdatePrimaryRegion` la région où la clé multirégionale est passée d'une clé de réplique à une clé primaire (ap-northeast-1). Cette entrée de journal n'identifie pas la région principale précédente.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice",
            "invokedBy": "kms.amazonaws.com"
    },
    "eventTime": "2021-03-10T20:23:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdatePrimaryRegion",
    "awsRegion": "ap-northeast-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "primaryRegion": "ap-northeast-1"
    },
    "responseElements": {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "ee408f36-ea01-422b-ac14-b0f147c68334",
    "eventID": "091e6be5-737f-43c6-8431-e3679d6d0619",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333"
}
```

VerifyMac

L'exemple suivant montre une entrée de AWS CloudTrail journal pour l'VerifyMacopération.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
     },
    "eventTime": "2022-03-31T19:25:54Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "VerifyMac",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "macAlgorithm": "HMAC_SHA_384",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "requestID": "f35da560-edff-4d6e-9b40-fb306fa9ef1e",
    "eventID": "6b464487-6dea-44cd-84ad-225d7450c975",
    "readOnly": true,
    "resources": [
        {
           "accountId": "111122223333",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Vérification

Ces exemples montrent les entrées du AWS CloudTrail journal pour l'opération Verify.

L'exemple suivant montre une entrée de CloudTrail journal pour une opération Verify qui utilise une clé RSA KMS asymétrique pour vérifier une signature numérique.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-03-07T22:50:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Verify",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "signingAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256",
        "keyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "messageType": "RAW"
    },
    "responseElements": null,
    "requestID": "c73ab82a-af82-4750-ae2c-b6bb790e9c28",
    "eventID": "3b4331cd-5b7b-4de5-bf5f-82ec22f0dac0",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
```

```
    "eventCategory": "Management"
}
```

Premier EC2 exemple d'Amazon

L'exemple suivant enregistre la création par un principal IAM d'un volume chiffré à l'aide de la clé de volume par défaut dans la console EC2 de gestion Amazon.

L'exemple suivant montre une entrée de CloudTrail journal dans laquelle l'utilisateur Alice crée un volume chiffré avec une clé de volume par défaut dans la console EC2 de gestion Amazon. L'enregistrement du fichier EC2 journal inclut un `volumeId` champ dont la valeur est de`"vol-13439757"`. L' AWS KMS enregistrement contient un `encryptionContext` champ dont la valeur est de`"aws:ebs:id": "vol-13439757"`. De même, les identificateurs `principalId` et `accountId` entre les deux enregistrements correspondent. Les enregistrements reflètent le fait que la création d'un volume chiffré génère une clé de données qui est utilisée pour chiffrer le contenu du volume.

```
{
  "Records": [
     {
      "eventVersion": "1.02",
      "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
            "userName": "Alice"
        },
      "eventTime": "2014-11-05T20:50:18Z",
      "eventSource": "ec2.amazonaws.com",
      "eventName": "CreateVolume",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "AWS Internal",
      "requestParameters": {
        "size": "10",
        "zone": "us-east-1a",
        "volumeType": "gp2",
        "encrypted": true
      },
      "responseElements": {
```

```
        "volumeId": "vol-13439757",
        "size": "10",
        "zone": "us-east-1a",
        "status": "creating",
        "createTime": 1415220618876,
        "volumeType": "gp2",
        "iops": 30,
        "encrypted": true
      },
      "requestID": "1565210e-73d0-4912-854c-b15ed349e526",
      "eventID": "a3447186-135f-4b00-8424-bc41f1a93b4f",
      "eventType": "AwsApiCall",
      "recipientAccountId": "123456789012"
    },
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
      },
      "eventTime": "2014-11-05T20:50:19Z",
      "eventSource": "kms.amazonaws.com",
      "eventName": "GenerateDataKeyWithoutPlaintext",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "&AWS; Internal",
      "requestParameters": {
        "encryptionContext": {
          "aws:ebs:id": "vol-13439757"
        },
        "numberOfBytes": 64,
        "keyId": "alias/aws/ebs"
      },
      "responseElements": null,
      "requestID": "create-123456789012-758241111-1415220618",
      "eventID": "4bd2a696-d833-48cc-b72c-05e61b608399",
      "readOnly": true,
      "resources": [
        {
```

```
          "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "accountId": "111122223333"
        }
      ],
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

EC2 Exemple 2 d'Amazon

Dans l'exemple suivant, un directeur IAM exécutant une EC2 instance Amazon crée et monte un volume de données chiffré sous une clé KMS. Cette action génère plusieurs enregistrements de CloudTrail journal.

Lorsque le volume est créé, Amazon EC2, agissant pour le compte du client, obtient une clé de données cryptée auprès de AWS KMS (`GenerateDataKeyWithoutPlaintext`). Ensuite, il crée un octroi (`CreateGrant`) qui lui permet de déchiffrer la clé de données. Lorsque le volume est monté, Amazon EC2 appelle AWS KMS pour déchiffrer la clé de données (`Decrypt`).

Le `instanceId` de l' EC2 instance Amazon apparaît dans l'`RunInstances`événement. `"i-81e2f56c"` Le même ID d'instance qualifie le `granteePrincipal` de l'octroi créé (`"111122223333:aws:ec2-infrastructure:i-81e2f56c"`) et le rôle supposé qui est le principal dans l'appel `Decrypt` (`"arn:aws:sts::111122223333:assumed-role/aws:ec2-infrastructure/i-81e2f56c"`).

La clé ARN de la clé KMS qui protège le volume de données apparaît dans les trois AWS KMS appels (`CreateGrant``GenerateDataKeyWithoutPlaintext`, et`Decrypt`). `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`

```
{
  "Records": [
    {
      "eventVersion": "1.02",
      "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
```

```
            "userName": "Alice"
        },
        "eventTime": "2014-11-05T21:35:27Z",
        "eventSource": "ec2.amazonaws.com",
        "eventName": "RunInstances",
        "awsRegion": "us-west-2",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "AWS Internal",
        "requestParameters": {
          "instancesSet": {
            "items": [
              {
                "imageId": "ami-b66ed3de",
                "minCount": 1,
                "maxCount": 1
              }
            ]
          },
          "groupSet": {
            "items": [
              {
                "groupId": "sg-98b6e0f2"
              }
            ]
          },
          "instanceType": "m3.medium",
          "blockDeviceMapping": {
            "items": [
              {
                "deviceName": "/dev/xvda",
                "ebs": {
                  "volumeSize": 8,
                  "deleteOnTermination": true,
                  "volumeType": "gp2"
                }
              },
              {
                "deviceName": "/dev/sdb",
                "ebs": {
                  "volumeSize": 8,
                  "deleteOnTermination": false,
                  "volumeType": "gp2",
                  "encrypted": true
                }
```

```
            }
          ]
        },
        "monitoring": {
          "enabled": false
        },
        "disableApiTermination": false,
        "instanceInitiatedShutdownBehavior": "stop",
        "clientToken": "XdKUT141516171819",
        "ebsOptimized": false
      },
      "responseElements": {
        "reservationId": "r-5ebc9f74",
        "ownerId": "111122223333",
        "groupSet": {
          "items": [
            {
              "groupId": "sg-98b6e0f2",
              "groupName": "launch-wizard-2"
            }
          ]
        },
        "instancesSet": {
          "items": [
            {
              "instanceId": "i-81e2f56c",
              "imageId": "ami-b66ed3de",
              "instanceState": {
                "code": 0,
                "name": "pending"
              },
              "amiLaunchIndex": 0,
              "productCodes": {

              },
              "instanceType": "m3.medium",
              "launchTime": 1415223328000,
              "placement": {
                "availabilityZone": "us-east-1a",
                "tenancy": "default"
              },
              "monitoring": {
                "state": "disabled"
              },
```

```
            "stateReason": {
              "code": "pending",
              "message": "pending"
            },
            "architecture": "x86_64",
            "rootDeviceType": "ebs",
            "rootDeviceName": "/dev/xvda",
            "blockDeviceMapping": {

            },
            "virtualizationType": "hvm",
            "hypervisor": "xen",
            "clientToken": "XdKUT1415223327917",
            "groupSet": {
              "items": [
                {
                  "groupId": "sg-98b6e0f2",
                  "groupName": "launch-wizard-2"
                }
              ]
            },
            "networkInterfaceSet": {

            },
            "ebsOptimized": false
          }
        ]
      }
    },
    "requestID": "41c4b4f7-8bce-4773-bf0e-5ae3bb5cbce2",
    "eventID": "cd75a605-2fee-4fda-b847-9c3d330ebaae",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  },
  {
    "eventVersion": "1.02",
    "userIdentity": {
          "type": "IAMUser",
          "principalId": "EX_PRINCIPAL_ID",
          "arn": "arn:aws:iam::111122223333:user/Alice",
          "accountId": "111122223333",
          "accessKeyId": "EXAMPLE_KEY_ID",
          "userName": "Alice"
    },
```

```
      "eventTime": "2014-11-05T21:35:35Z",
      "eventSource": "kms.amazonaws.com",
      "eventName": "CreateGrant",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "AWS Internal",
      "requestParameters": {
        "constraints": {
          "encryptionContextSubset": {
            "aws:ebs:id": "vol-f67bafb2"
          }
        },
        "granteePrincipal": "111122223333:aws:ec2-infrastructure:i-81e2f56c",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      },
      "responseElements": {
        "grantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a"
      },
      "requestID": "41c4b4f7-8bce-4773-bf0e-5ae3bb5cbce2",
      "eventID": "c1ad79e3-0d3f-402a-b119-d5c31d7c6a6c",
      "readOnly": false,
      "resources": [
        {
          "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "accountId": "111122223333"
        }
      ],
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    },
    {
      "eventVersion": "1.02",
      "userIdentity": {
          "type": "IAMUser",
          "principalId": "EX_PRINCIPAL_ID",
          "arn": "arn:aws:iam::111122223333:user/Alice",
          "accountId": "111122223333",
          "accessKeyId": "EXAMPLE_KEY_ID",
          "userName": "Alice"
      },
      "eventTime": "2014-11-05T21:35:32Z",
      "eventSource": "kms.amazonaws.com",
```

```
      "eventName": "GenerateDataKeyWithoutPlaintext",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "AWS Internal",
      "requestParameters": {
        "encryptionContext": {
          "aws:ebs:id": "vol-f67bafb2"
        },
        "numberOfBytes": 64,
        "keyId": "alias/aws/ebs"
      },
      "responseElements": null,
      "requestID": "create-111122223333-758247346-1415223332",
      "eventID": "ac3cab10-ce93-4953-9d62-0b6e5cba651d",
      "readOnly": true,
      "resources": [
        {
          "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "accountId": "111122223333"
        }
      ],
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    },
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "111122223333:aws:ec2-infrastructure:i-81e2f56c",
        "arn": "arn:aws:sts::111122223333:assumed-role/aws:ec2-infrastructure/
i-81e2f56c",
        "accountId": "111122223333",
        "accessKeyId": "",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2014-11-05T21:35:38Z"
          },
          "sessionIssuer": {
            "type": "Role",
            "principalId": "111122223333:aws:ec2-infrastructure",
            "arn": "arn:aws:iam::111122223333:role/aws:ec2-infrastructure",
            "accountId": "111122223333",
```

```
            "userName": "aws:ec2-infrastructure"
          }
        }
      },
      "eventTime": "2014-11-05T21:35:47Z",
      "eventSource": "kms.amazonaws.com",
      "eventName": "Decrypt",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.2.0",
      "requestParameters": {
        "encryptionContext": {
          "aws:ebs:id": "vol-f67bafb2"
        }
      },
      "responseElements": null,
      "requestID": "b4b27883-6533-11e4-b4d9-751f1761e9e5",
      "eventID": "edb65380-0a3e-4123-bbc8-3d1b7cff49b0",
      "readOnly": true,
      "resources": [
        {
          "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "accountId": "111122223333"
        }
      ],
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```


Surveillez les clés KMS avec Amazon CloudWatch

Vous pouvez suivre votre AWS KMS keys utilisation d'Amazon CloudWatch, un AWS service qui collecte et traite les données brutes pour AWS KMS en faire des indicateurs lisibles en temps quasi réel. Ces données sont enregistrées pour une durée de deux semaines pour que vous puissiez accéder aux informations historiques, et mieux comprendre l'utilisation de vos clés KMS et leur évolution au fil du temps.

Vous pouvez utiliser Amazon CloudWatch pour vous avertir d'événements importants, tels que les suivants.

- Les éléments de clé importés dans une clé KMS approchent de leur date d'expiration.
- Une clé KMS en attente de suppression est toujours utilisée.
- Les éléments de clé dans une clé KMS ont effectué automatiquement une rotation.
- Une clé KMS a été supprimée.

Vous pouvez également créer une CloudWatch alarme Amazon qui vous avertit lorsque le taux de demandes atteint un certain pourcentage de la valeur du quota. Pour plus de détails, consultez Gérer vos taux de demandes AWS KMS d'API à l'aide de Service Quotas et d'Amazon CloudWatch dans le blog sur la AWS sécurité.

AWS KMS métriques et dimensions

AWS KMS prédéfinit CloudWatch les métriques Amazon pour vous permettre de surveiller plus facilement les données critiques et de créer des alarmes. Vous pouvez consulter les AWS KMS statistiques à l'aide de l' CloudWatch API AWS Management Console et de l'Amazon.

Cette section répertorie chaque AWS KMS métrique et les dimensions de chaque métrique, et fournit des conseils de base pour créer des CloudWatch alarmes basées sur ces métriques et dimensions.

> **Note**
>
> Nom du groupe de dimensions :
> Pour afficher une métrique dans la CloudWatch console Amazon, dans la section Mesures, sélectionnez le nom du groupe de dimensions. Vous pouvez ensuite filtrer en fonction du Metric name (Nom de la métrique). Cette rubrique inclut le nom de la métrique et le nom du groupe de dimensions pour chaque métrique AWS KMS .

Vous pouvez consulter AWS KMS les statistiques à l'aide de l' CloudWatch API AWS Management Console et de l'Amazon. Pour plus d'informations, consultez la section Afficher les statistiques disponibles dans le guide de CloudWatch l'utilisateur Amazon.

Rubriques

- SuccessfulRequest
- SecondsUntilKeyMaterialExpiration
- Nuage HSMKey StoreThrottle
- ExternalKeyStoreThrottle

- XksProxyCertificateDaysToExpire
- XksProxyCredentialAge
- XksProxyErrors
- XksExternalKeyManagerStates
- XksProxyLatency

SuccessfulRequest

Nombre de demandes d'opérations cryptographiques réussies sur une clé KMS spécifique. En utilisant cette `SuccessfulRequest` métrique, vous pouvez appliquer un filtrage au niveau des clés à l'utilisation de AWS KMS l'API dans. CloudWatch La `Sum` statistique de cette métrique définit le nombre total de demandes réussies au cours de la période.

Utilisez cette métrique pour identifier les clés KMS qui consomment la plus grande partie de votre quota de demandes ou qui contribuent le plus aux frais d'API. Vous pouvez également créer une CloudWatch alarme basée sur la `SuccesfulRequest` métrique pour vous avertir des modèles d'utilisation anormaux de l' AWS KMS API. Ces alertes peuvent aider à identifier les flux de travail inefficaces susceptibles de dépasser involontairement vos quotas de demandes ou d'entraîner des frais imprévus.

Dimensions pour **`SuccessfulRequest`**

Dimension	Description
KeyArn	Valeur pour chaque clé KMS.
Opération	Valeur pour chaque opération AWS KMS d'API. Cette métrique s'applique uniquement aux opérations cryptographiques.

Pour les ReEncryptopérations, la `SuccessfulRequest` métrique inclut des dimensions pour les clés KMS source et de destination.

Dimension	Description
SourceKey Arn	Valeur de la clé KMS qui a déchiffré le texte chiffré.

Dimension	Description
DestinationKeyArn	Valeur de la clé KMS qui a rechiffré les données.
Opération	Valeur pour chaque opération AWS KMS d'API, dans ce cas, ReEncrypt.

SecondsUntilKeyMaterialExpiration

Nombre de secondes restant avant l'expiration du contenu clé importé d'une clé KMS dont la date d'expiration est la plus proche. Cette métrique n'est valide que pour les clés KMS avec des éléments de clé importés (dont l'origine des éléments de clé est `EXTERNAL`) et une date d'expiration.

Utilisez cette métrique pour savoir combien de temps il reste avant l'expiration de votre matériel clé importé expirant le plus tôt. Lorsque ce délai tombe en dessous d'un seuil que vous définissez, vous devez réimporter le matériel clé avec une nouvelle date d'expiration pour que la clé KMS reste utilisable. La métrique `SecondsUntilKeyMaterialExpiration` est spécifique à une clé KMS. Vous ne pouvez pas utiliser cette métrique pour surveiller plusieurs clés KMS ou les clés KMS que vous pourriez créer ultérieurement. Pour obtenir de l'aide sur la création CloudWatch d'une alarme pour surveiller cette métrique, consultezCréer une CloudWatch alarme en cas d'expiration du matériel clé importé.

`Minimum` est la statistique la plus utile pour cette métrique. Elle indique le plus petit temps restant pour tous les points de données de la période statistique spécifiée. La seule unité valide pour cette métrique est `Seconds`.

Nom du groupe de dimensions : Per-Key Metrics (Métriques par clé)

Dimensions pour **`SecondsUntilKeyMaterialExpiration`**

Dimension	Description ; en rapport avec AWS
KeyId	Valeur pour chaque clé KMS.

Lorsque vous planifiez la suppression d'une clé KMS, AWS KMS applique une période d'attente avant de supprimer la clé KMS. Vous pouvez utiliser la période d'attente pour vous assurer de ne pas avoir besoin de la clé KMS maintenant ni par la suite. Vous pouvez également configurer une CloudWatch alarme pour vous avertir si une personne ou une application tente d'utiliser la clé KMS

lors d'une opération cryptographique pendant la période d'attente. Si vous recevez une notification de ce type d'alarme, vous pouvez annuler la suppression de la clé KMS.

Pour obtenir des instructions, veuillez consulter Créez une alarme qui détecte l'utilisation d'une clé KMS en attente de suppression.

Nuage HSMKey StoreThrottle

Nombre de demandes d'opérations cryptographiques sur des clés KMS dans chaque magasin de AWS CloudHSM clés qui AWS KMS limite (répond par un). `ThrottlingException` Cette métrique s'applique uniquement aux magasins AWS CloudHSM clés.

La `CloudHSMKeyStoreThrottle` métrique s'applique uniquement aux clés KMS d'un magasin de AWS CloudHSM clés et uniquement aux demandes d'opérations cryptographiques. AWS KMS limite ces demandes lorsque le taux de demandes dépasse le quota de demandes de stockage de clés personnalisé pour votre magasin de AWS CloudHSM clés. Cette métrique inclut également la régulation par le AWS CloudHSM cluster.

Nom du groupe de dimensions : Keystore Throttle Metrics (Métriques de limitation du magasin de clés)

Dimension	Description
CustomKey StoreId	Valeur pour chaque magasin de AWS CloudHSM clés.
KmsOperation	Valeur pour chaque opération AWS KMS d'API. Cette métrique s'applique uniquement aux opérations cryptographiques sur les clés KMS dans un magasin de AWS CloudHSM clés.
KeySpec	Valeur pour chaque type de clé KMS. La seule spécification de clé prise en charge pour les clés KMS dans un magasin de clés est AWS CloudHSM SYMMETRIC _DEFAULT.

ExternalKeyStoreThrottle

Nombre de demandes d'opérations cryptographiques sur des clés KMS dans chaque magasin de clés externe qui AWS KMS limite (répond par un). `ThrottlingException` Cette métrique ne s'applique qu'aux magasins de clés externes.

La `ExternalKeyStoreThrottle` métrique s'applique uniquement aux clés KMS dans un magasin de clés externe et uniquement aux demandes d'opérations cryptographiques. AWS KMS limite ces demandes lorsque le taux de demandes dépasse le quota de demandes de stockage de clés personnalisé pour votre magasin de clés externe. Cette métrique n'inclut pas la limitation par votre proxy de magasin de clés externe ou votre gestionnaire de clés externe.

Utilisez cette métrique pour revoir et ajuster la valeur de votre quota de demandes personnalisé en magasin de clés. Si cette métrique indique que AWS KMS vos demandes pour ces clés KMS sont fréquemment limitées, vous pouvez envisager de demander une augmentation de la valeur du quota de demandes de stockage de clés personnalisé. Si vous avez besoin d'aide, consultez Requesting a quota increase (Demande d'augmentation de quota) dans le Guide de l'utilisateur Service Quotas.

Si vous obtenez très fréquemment des erreurs `KMSInvalidStateException` avec un message expliquant que la requête a été rejetée « en raison d'un taux de requêtes très élevé » ou que la requête a été rejetée « car le proxy de magasin de clés externe n'a pas répondu à temps », cela peut indiquer que votre gestionnaire de clés externe ou votre proxy de magasin de clés externe ne peut pas suivre le rythme du taux de requêtes actuel. Si possible, réduisez votre taux de requêtes. Vous pouvez également envisager de demander une diminution de la valeur de votre quota de requêtes du magasin de clés personnalisé. La diminution de cette valeur de quota peut augmenter la régulation (et la valeur `ExternalKeyStoreThrottle` métrique), mais cela indique que les demandes excédentaires AWS KMS sont rejetées rapidement avant qu'elles ne soient envoyées à votre proxy de stockage de clés externe ou à votre gestionnaire de clés externe. Pour solliciter une réduction de quota, accédez au Centre AWS Support et créez une demande.

Nom du groupe de dimensions : Keystore Throttle Metrics (Métriques de limitation du magasin de clés)

Dimension	Description
CustomKeyStoreId	Valeur pour chaque magasin de clés externe.
KmsOperation	Valeur pour chaque opération AWS KMS d'API. Cette métrique s'applique uniquement aux opérations cryptographiques sur les clés KMS dans un magasin de clés externe.
KeySpec	Valeur pour chaque type de clé KMS. La seule spécification de clé prise en charge pour les clés KMS dans un magasin de clés externe est SYMMETRIC_DEFAULT.

XksProxyCertificateDaysToExpire

Nombre de jours avant l'expiration du certificat TLS de votre point de terminaison du proxy de magasin de clés externe (`XksProxyUriEndpoint`). Cette métrique ne s'applique qu'aux magasins de clés externes.

Utilisez cette métrique pour créer une CloudWatch alarme qui vous avertit de l'expiration prochaine de votre certificat TLS. Lorsque le certificat expire, AWS KMS impossible de communiquer avec le proxy de stockage de clés externe. Toutes les données protégées par des clés KMS dans votre magasin de clés externe deviennent inaccessibles jusqu'à ce que vous renouveliez le certificat.

Une alerte de certificat informe de l'expiration d'un certificat qui pourrait vous empêcher d'accéder à vos ressources chiffrées. Réglez l'alerte pour donner à votre organisation le temps de renouveler le certificat avant qu'il n'expire.

Nom du groupe de dimensions : XKS Proxy Certificate Metrics (Métriques du certificat du proxy XKS)

Dimension	Description
CustomKeyStoreId	Valeur pour chaque magasin de clés externe.
CertificateName	Nom du sujet (CN) dans le certificat TLS.

Vous pouvez créer des CloudWatch alarmes en fonction des métriques des banques de clés externes et des clés KMS des banques de clés externes. Pour obtenir des instructions, veuillez consulter Surveillez les magasins de clés externes.

XksProxyCredentialAge

Nombre de jours écoulés depuis que les informations d'identification pour l'authentification de proxy (`XksProxyAuthenticationCredential`) du magasin de clés externe actuel ont été associées au magasin de clés externe. Ce décompte commence lorsque vous saisissez les informations d'identification pour l'authentification dans le cadre de la création ou de la mise à jour de votre magasin de clés externe. Cette métrique ne s'applique qu'aux magasins de clés externes.

Cette valeur est conçue pour vous rappeler l'âge de vos informations d'identification pour l'authentification. Toutefois, étant donné que nous commençons le décompte lorsque vous associez

les informations d'identification à votre magasin de clés externe, et non lorsque vous créez vos informations d'identification pour l'authentification sur le proxy de votre magasin de clés externe, cela peut ne pas être un indicateur précis de l'âge des informations d'identification sur le proxy.

Utilisez cette métrique pour créer une CloudWatch alarme qui vous rappelle de changer vos informations d'identification d'authentification du proxy de stockage de clés externe.

Nom du groupe de dimensions : Per-Keystore Metrics (Métriques par magasin de clés)

Dimension	Description
CustomKeyStoreId	Valeur pour chaque magasin de clés externe.

Vous pouvez créer des CloudWatch alarmes en fonction des métriques des banques de clés externes et des clés KMS des banques de clés externes. Pour obtenir des instructions, veuillez consulter Surveillez les magasins de clés externes.

XksProxyErrors

Le nombre d'exceptions liées aux AWS KMS demandes adressées à votre proxy de stockage de clés externe. Ce décompte inclut les exceptions auxquelles le proxy de stockage de clés externe revient AWS KMS et les erreurs de délai d'expiration qui se produisent lorsque le proxy de stockage de clés externe ne répond pas AWS KMS dans l'intervalle de 250 millisecondes. Cette métrique ne s'applique qu'aux magasins de clés externes.

Utilisez cette métrique pour effectuer le suivi du taux d'erreurs des clés KMS dans votre magasin de clés externe. Elle révèle les erreurs les plus fréquentes, ce qui vous permet de hiérarchiser vos efforts d'ingénierie. Par exemple, les clés KMS qui génèrent des taux élevés d'erreurs non récupérables peuvent indiquer un problème de configuration de votre magasin de clés externe. Pour consulter la configuration de votre magasin de clés externe, veuillez consulter la rubrique Afficher les magasins de clés externes. Pour modifier les paramètres de votre clé externe, veuillez consulter la rubrique Modifier les propriétés du magasin de clés externe.

Nom du groupe de dimensions : XKS Proxy Error Metrics (Métriques d'erreurs du proxy XKS)

Dimension	Description
CustomKeyStoreId	Valeur pour chaque magasin de clés externe.
KmsOperation	Valeur pour chaque opération AWS KMS d'API qui a généré une demande au proxy XKS.
XksOperation	Valeur pour chaque opération de l'API du proxy de magasin de clés externe.
KeySpec	Valeur pour chaque type de clé KMS. La seule spécification de clé prise en charge pour les clés KMS dans un magasin de clés externe est SYMMETRIC_DEFAULT.
ErrorType	Valeurs : • Erreurs récupérables : susceptibles d'être transitoires, telles que des erreurs de mise en réseau. • Erreurs non récupérables : susceptibles d'indiquer un problème avec la configuration du magasin de clés personnalisé ou des composants externes. • N/A : requête réussie ; aucune erreur
ExceptionName	Valeurs : • Nom de l'exception • Aucune : requête réussie ; aucune erreur

Vous pouvez créer des CloudWatch alarmes en fonction des métriques des banques de clés externes et des clés KMS des banques de clés externes. Pour obtenir des instructions, veuillez consulter Surveillez les magasins de clés externes.

XksExternalKeyManagerStates

Décompte du nombre d'instances de gestionnaire de clés externe dans chacun des états suivants : `Active`, `Degraded` et `Unavailable`. Les informations relatives à cette métrique proviennent du proxy de magasin de clés externe associé à chaque magasin de clés externe. Cette métrique ne s'applique qu'aux magasins de clés externes.

Les états des instances de gestionnaire de clés externe associées à un magasin de clés externe sont les suivants. Chaque proxy de magasin de clés externe peut utiliser des indicateurs différents pour mesurer l'état de votre gestionnaire de clés externe. Pour plus de détails, veuillez consulter la documentation de votre proxy de magasin de clés externe.

- `Active` : le gestionnaire de clés externe est sain.
- `Degraded` : le gestionnaire de clés externe est défectueux, mais il peut toujours traiter le trafic.
- `Unavailable` : le gestionnaire de clés externe ne peut pas traiter le trafic.

Utilisez cette métrique pour créer une CloudWatch alarme qui vous avertit en cas de dégradation ou d'indisponibilité d'instances de gestionnaire de clés externes. Pour déterminer quelles instances de gestionnaire de clés externe se trouvent dans chaque état, consultez les journaux du proxy de votre magasin de clés externe.

Nom du groupe de dimensions : XKS External Key Manager Metrics (Métriques du gestionnaire de clés externe XKS)

Dimension	Description
CustomKeyStoreId	Valeur pour chaque magasin de clés externe.
XksExternalKeyManagerState	Valeur pour chaque état.

Vous pouvez créer des CloudWatch alarmes en fonction des métriques des banques de clés externes et des clés KMS des banques de clés externes. Pour obtenir des instructions, veuillez consulter Surveillez les magasins de clés externes.

XksProxyLatency

Nombre de millisecondes nécessaires à un proxy de magasin de clés externe pour répondre à une requête AWS KMS . Si le délai de la requête a expiré, la valeur enregistrée est la limite de délai d'expiration de 250 millisecondes. Cette métrique ne s'applique qu'aux magasins de clés externes.

Utilisez cette métrique pour évaluer les performances de votre proxy de magasin de clés externe et de votre gestionnaire de clés externe. Par exemple, si le proxy expire fréquemment lors des opérations de chiffrement et de déchiffrement, consultez votre administrateur de proxy externe.

Les réponses lentes peuvent également indiquer que votre gestionnaire de clés externe ne peut pas gérer le trafic de demandes actuel. AWS KMS recommande que votre gestionnaire de clés externe soit capable de traiter jusqu'à 1 800 demandes d'opérations cryptographiques par seconde. Si votre gestionnaire de clés externe ne peut pas gérer le taux de 1 800 requêtes par seconde, pensez à demander une diminution de votre quota de requêtes de clés KMS dans un magasin de clés personnalisé. Les requêtes d'opérations cryptographiques utilisant les clés KMS de votre magasin de clés externe échoueront rapidement, avec une exception de limitation, au lieu d'être traitées puis rejetées par le proxy de votre magasin de clés externe ou le gestionnaire de clés externe.

Nom du groupe de dimensions : XKS Proxy Latency Metrics (Métriques de latence de proxy XKS)

Dimension	Description
CustomKeyStoreId	Valeur pour chaque magasin de clés externe.
KmsOperation	Valeur pour chaque opération AWS KMS d'API qui a généré une demande au proxy XKS.
XksOperation	Valeur pour chaque opération de l'API du proxy de magasin de clés externe.
KeySpec	Valeur pour chaque type de clé KMS. La seule spécification de clé prise en charge pour les clés KMS dans un magasin de clés externe est SYMMETRIC_DEFAULT.

Vous pouvez créer des CloudWatch alarmes en fonction des métriques des banques de clés externes et des clés KMS des banques de clés externes. Pour obtenir des instructions, veuillez consulter Surveillez les magasins de clés externes.

Créer une CloudWatch alarme en cas d'expiration du matériel clé importé

Vous pouvez créer une CloudWatch alarme qui vous avertit lorsque le contenu clé importé d'une clé KMS approche de son expiration. Par exemple, l'alarme peut vous notifier lorsque le délai d'expiration est inférieur à 30 jours.

Lorsque vous importez des éléments de clé dans une clé KMS, vous pouvez éventuellement spécifier une date et heure à laquelle les éléments de clé doivent expirer. Lorsque le contenu clé expire, il est AWS KMS supprimé et la clé KMS devient inutilisable. Pour utiliser la clé KMS à nouveau, vous devez réimporter les éléments de clé. Toutefois, si vous réimportez les éléments de clé avant qu'ils n'expirent, vous pouvez éviter de perturber les processus qui utilisent cette clé KMS.

Cette alarme utilise la `SecondsUntilKeyMaterialExpires`métrique AWS KMS publiée sur CloudWatch pour les clés KMS dont le contenu clé importé expire. Chaque alarme utilise cette métrique pour surveiller les éléments de clé importés pour une clé KMS particulière. Vous ne pouvez pas créer une alarme unique pour toutes les clés KMS dont les éléments de clé expire ou une alarme pour les clés KMS que vous pourriez créer ultérieurement.

Prérequis

Les ressources suivantes sont requises pour une CloudWatch alarme qui surveille l'expiration du matériel clé importé.

- Une clé KMS dont les éléments de clé importés expirent.
- Une rubrique Amazon SNS Pour plus de détails, consultez la rubrique Création d'un Amazon SNS dans le guide de CloudWatch l'utilisateur Amazon.

Créez l'alarme

Suivez les instructions de la section Création CloudWatch d'une alarme basée sur un seuil statique en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Sélectionner une métrique	Choisissez KMS, puis choisissez Per-Key Metrics. Choisissez la ligne contenant la clé KMS et la métrique `SecondsUntilKeyMaterialExpires` . Ensuite, choisissez Select metric (Sélectionner une métrique). La liste Métriques affiche les métriques `SecondsUntilKeyMaterialExpires` uniquement pour les clés KMS dont les éléments de clé importés expirent. Si vous ne disposez pas de clés KMS avec ces propriétés dans le compte et la région, cette liste est vide.

Champ	Valeur
Statistique	Minimum
Période	1 minute
Type de seuil	Statique
Chaque fois que …	Chaque fois que *metric-name* c'est supérieur à 1

Créez des CloudWatch alarmes pour les magasins de clés externes

Vous pouvez créer des CloudWatch alarmes Amazon en fonction des statistiques externes du magasin clé pour vous avertir lorsqu'une valeur métrique dépasse le seuil que vous avez spécifié. L'alarme peut envoyer le message à une rubrique Amazon Simple Notification Service (Amazon SNS) ou à une politique Amazon EC2Auto Scaling. Pour obtenir des informations détaillées sur les CloudWatch alarmes, consultez la section Utilisation des CloudWatch alarmes Amazon dans le guide de CloudWatch l'utilisateur Amazon.

Avant de créer une CloudWatch alarme Amazon, vous avez besoin d'une rubrique Amazon SNS. Pour plus de détails, consultez la rubrique Création d'un Amazon SNS dans le guide de CloudWatch l'utilisateur Amazon.

Rubriques

- Création d'une alarme pour l'expiration du certificat
- Création d'une alarme pour l'expiration du délai de réponse
- Créez une alarme pour les erreurs réessayables
- Créez une alarme pour les erreurs non réessayables

Création d'une alarme pour l'expiration du certificat

Cette alarme utilise la XksProxyCertificateDaysToExpire métrique AWS KMS publiée sur CloudWatch pour enregistrer l'expiration prévue du certificat TLS associé à votre point de terminaison proxy de stockage de clés externe. Vous ne pouvez pas créer une alerte unique pour tous les magasins de clés externes de votre compte ou une alerte pour les magasins de clés externes que vous pourriez créer à l'avenir.

Nous vous recommandons de régler l'alerte pour qu'elle vous avertisse 10 jours avant l'expiration de votre certificat, mais vous êtes invité à définir le seuil qui correspond le mieux à vos besoins.

Créez l'alarme

Suivez les instructions de la section Création CloudWatch d'une alarme basée sur un seuil statique en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Sélectionner une métrique	Choisissez KMS, puis choisissez XKS Proxy Certificate Metrics (Métriques de certificat de proxy XKS). Cochez la case à côté du `XksProxyCertificateName` que vous souhaitez surveiller. Ensuite, choisissez Select metric (Sélectionner une métrique).
Statistique	Minimum
Période	5 minutes
Type de seuil	Statique
Chaque fois que …	À chaque fois `Lower` que XksProxyCertificateDaysToExpirec'est le cas`10`.

Création d'une alarme pour l'expiration du délai de réponse

Cette alarme utilise la XksProxyLatency métrique AWS KMS publiée sur CloudWatch pour enregistrer le nombre de millisecondes nécessaires à un proxy de stockage de clés externe pour répondre à une demande. AWS KMS Vous ne pouvez pas créer une alerte unique pour tous les magasins de clés externes de votre compte ou une alerte pour les magasins de clés externes que vous pourriez créer à l'avenir.

AWS KMS attend du proxy de stockage de clés externe qu'il réponde à chaque demande dans un délai de 250 millisecondes. Nous vous recommandons de configurer une alerte pour vous avertir lorsque le proxy de votre magasin de clés externe met plus de 200 millisecondes à répondre, mais vous êtes invité à définir le seuil qui correspond le mieux à vos besoins.

Créez l'alarme

Suivez les instructions de la section Création CloudWatch d'une alarme basée sur un seuil statique en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Sélectionner une métrique	Choisissez KMS, puis choisissez XKS Proxy Latency Metrics (Métriques de latence de proxy XKS). Cochez la case à côté du `KmsOperation` que vous souhaitez surveiller. Ensuite, choisissez Select metric (Sélectionner une métrique).
Statistique	Moyenne
Période	5 minutes
Type de seuil	Statique
Chaque fois que …	À chaque fois `Greater` que XksProxyLatencyc'est le cas`200`.

Créez une alarme pour les erreurs réessayables

Cette alarme utilise la XksProxyErrors métrique AWS KMS publiée sur CloudWatch pour enregistrer le nombre d'exceptions liées aux AWS KMS demandes adressées à votre proxy de stockage de clés externe. Vous ne pouvez pas créer une alerte unique pour tous les magasins de clés externes de votre compte ou une alerte pour les magasins de clés externes que vous pourriez créer à l'avenir.

Les erreurs récupérables réduisent votre pourcentage de fiabilité et peuvent indiquer des erreurs réseau. Nous vous recommandons de définir une alerte pour vous avertir lorsque plus de cinq erreurs récupérables sont enregistrées sur une période d'une minute, mais vous êtes invité à définir le seuil qui correspond le mieux à vos besoins.

Suivez les instructions de la section Création CloudWatch d'une alarme basée sur un seuil statique en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Sélectionner une métrique	Choisissez l'onglet Query (Requête). Choisissez `AWS/KMS` comme Namespace (Espace de noms). Saisissez `SUM(XksProxyErrors)` comme Metric name (Nom de la métrique). Saisissez `ErrorType = Retryable` pour Filter by (Filtrer par). Cliquez sur Exécuter. Ensuite, choisissez Select metric (Sélectionner une métrique).
Étiquette	*`Retryable errors`*
Période	1 minute
Type de seuil	Statique
Chaque fois que …	Chaque fois que `q1` est `Greater` que `5`.

Créez une alarme pour les erreurs non réessayables

Cette alarme utilise la XksProxyErrors métrique AWS KMS publiée sur CloudWatch pour enregistrer le nombre d'exceptions liées aux AWS KMS demandes adressées à votre proxy de stockage de clés externe. Vous ne pouvez pas créer une alerte unique pour tous les magasins de clés externes de votre compte ou une alerte pour les magasins de clés externes que vous pourriez créer à l'avenir.

Les erreurs non récupérables peuvent indiquer un problème de configuration de votre magasin de clés externe. Nous vous recommandons de définir une alerte pour vous avertir lorsque plus de cinq erreurs non récupérables sont enregistrées sur une période d'une minute, mais vous êtes invité à définir le seuil qui correspond le mieux à vos besoins.

Suivez les instructions de la section Création CloudWatch d'une alarme basée sur un seuil statique en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Sélectionner une métrique	Choisissez l'onglet Query (Requête). Choisissez `AWS/KMS` comme Namespace (Espace de noms). Saisissez `SUM(XksProxyErrors)` comme Metric name (Nom de la métrique). Saisissez `ErrorType = Non-retryable` pour Filter by (Filtrer par). Cliquez sur Exécuter. Ensuite, choisissez Select metric (Sélectionner une métrique).
Étiquette	*`Non-retryable errors`*
Période	1 minute
Type de seuil	Statique
Chaque fois que …	Chaque fois que `q1` est `Greater` que `5`.

Surveillez les clés KMS avec Amazon EventBridge

Vous pouvez utiliser Amazon EventBridge (anciennement Amazon CloudWatch Events) pour vous avertir des événements importants suivants survenus dans le cycle de vie de vos clés KMS.

- Le contenu clé d'une clé KMS a fait l'objet d'une rotation automatique ou à la demande.
- Le matériel clé importé dans la clé KMS expirée
- Une clé KMS dont la suppression avait été planifiée a été supprimée.

AWS KMS s'intègre EventBridge à Amazon pour vous informer des événements importants qui affectent vos clés KMS. Chaque événement est représenté au format JSON (JavaScriptObject Notation) et inclut le nom de l'événement, la date et l'heure auxquelles l'événement s'est produit, ainsi que les événements concernés. Vous pouvez collecter ces événements et établir des règles qui les acheminent vers une ou plusieurs cibles, telles que les AWS Lambda fonctions, les rubriques

Amazon SNS, les files d'attente Amazon SQS, les flux dans Amazon Kinesis Data Streams ou les cibles intégrées.

Pour plus d'informations sur l'utilisation EventBridge avec d'autres types d'événements, notamment ceux émis AWS CloudTrail lors de l'enregistrement d'une demande d' read/write API, consultez le guide de EventBridge l'utilisateur Amazon.

Les rubriques suivantes décrivent les EventBridge événements AWS KMS générés.

Rotation de clé CMK dans KMS

AWS KMS prend en charge la rotation automatique et à la demande du contenu clé dans les clés KMS de chiffrement symétriques.

Chaque fois qu'il AWS KMS fait pivoter un matériau clé, il envoie un `KMS CMK Rotation` événement à EventBridge. AWS KMS génère cet événement dans la mesure du possible.

Voici un exemple de cet événement.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "KMS CMK Rotation",
  "source": "aws.kms",
  "account": "111122223333",
  "time": "2025-05-23T03:11:54Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  ],
  "detail": {
    "key-id": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "key-origin": "AWS_KMS",
    "rotation-type": "ON_DEMAND",
    "previous-key-material-id":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0",
    "current-key-material-id":
 "96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068"
  }
}
```

Expiration d'éléments de clé importés KMS

Lorsque vous importez des éléments de clé dans une clé KMS, vous pouvez éventuellement spécifier une heure à laquelle les éléments de clé doivent expirer. Lorsque le contenu clé expire, le AWS KMS supprime et envoie un `KMS Imported Key Material Expiration` événement correspondant à EventBridge. AWS KMS génère cet événement dans la mesure du possible.

Voici un exemple de cet événement.

```
{
  "version": "0",
  "id": "9da9af57-9253-4406-87cb-7cc400e43465",
  "detail-type": "KMS Imported Key Material Expiration",
  "source": "aws.kms",
  "account": "111122223333",
  "time": "2025-05-23T03:11:54Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  ],
  "detail": {
    "key-id": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "key-material-id":
 "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  }
}
```

Suppression d'une clé CMK dans KMS

Lorsque vous planifiez la suppression d'une clé KMS, AWS KMS applique une période d'attente avant de supprimer la clé KMS. Une fois la période d'attente terminée, AWS KMS supprime la clé KMS et envoie un `KMS CMK Deletion` événement à EventBridge. AWS KMS garantit cet EventBridge événement. En raison de nouvelles tentatives, il peut générer plusieurs événements en quelques secondes qui suppriment la même clé KMS.

Voici un exemple de cet événement.

```
{
  "version": "0",
  "id": "e9ce3425-7d22-412a-a699-e7a5fc3fbc9a",
  "detail-type": "KMS CMK Deletion",
```

```
  "source": "aws.kms",
  "account": "111122223333",
  "time": "2025-05-23T03:11:54Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  ],
  "detail": {
    "key-id": "1234abcd-12ab-34cd-56ef-1234567890ab"
  }
}
```

Alias dans AWS KMS

Un alias est un nom convivial pour une AWS KMS key. Par exemple, un alias vous permet de faire référence à une clé KMS en tant que `test-key` au lieu de `1234abcd-12ab-34cd-56ef-1234567890ab`.

Vous pouvez utiliser un alias pour identifier une clé KMS dans la AWS KMS console, dans l'DescribeKeyopération et dans les opérations cryptographiques, telles que Encrypt et. GenerateDataKey Les alias facilitent également la reconnaissance d'une Clé gérée par AWS. Les alias de ces clés KMS sont toujours au format `aws/<service-name>`. Par exemple, l'alias Clé gérée par AWS pour Amazon `aws/dynamodb` DynamoDB est. Vous pouvez établir des normes d'alias similaires pour vos projets, par exemple préfacer vos alias avec le nom d'un projet ou d'une catégorie.

Vous pouvez également autoriser et refuser l'accès aux clés KMS en fonction de leurs alias sans modifier les politiques ni gérer les octrois. Cette fonctionnalité fait partie de la AWS KMS prise en charge du contrôle d'accès basé sur les attributs (ABAC). Pour plus de détails, consultez Utiliser des alias pour contrôler l'accès aux clés KMS.

Une grande partie des performances des alias provient de votre capacité à modifier la clé KMS associée à un alias à tout moment. Les alias peuvent faciliter l'écriture et la maintenance de votre code. Par exemple, supposons que vous utilisiez un alias pour faire référence à une clé KMS particulière et que vous souhaitiez modifier la clé KMS. Dans ce cas, il suffit d'associer l'alias à une autre clé KMS. Vous n'avez pas besoin d'apporter de modifications à votre code.

Les alias facilitent également la réutilisation du même code dans différentes Régions AWS. Créez des alias portant le même nom dans plusieurs régions et associez chaque alias à une clé KMS dans sa région. Lorsque le code s'exécute dans chaque région, l'alias fait référence à la clé KMS associée dans cette région. Pour obtenir un exemple, consultez Apprenez à utiliser des alias dans vos applications.

Vous pouvez créer un alias pour une clé KMS dans la AWS KMS console, à l'aide de l'CreateAliasAPI ou du AWS::KMS::Alias AWS CloudFormation modèle.

L' AWS KMS API fournit un contrôle total des alias dans chaque compte et région. L'API inclut des opérations permettant de créer un alias (CreateAlias), d'afficher les noms d'alias et d'alias ARNs (ListAliases), de modifier la clé KMS associée à un alias (UpdateAlias) et de supprimer un alias (DeleteAlias).

Comment fonctionnent les alias

Découvrez comment les alias fonctionnent dans AWS KMS.

Un alias est une AWS ressource indépendante

Un alias n'est pas une propriété d'une clé KMS. Les actions que vous effectuez sur l'alias n'affectent pas sa clé KMS associée. Vous pouvez créer un alias pour une clé KMS, puis mettre à jour l'alias afin qu'il soit associé à une autre clé KMS. Vous pouvez même supprimer l'alias sans aucun effet sur la clé KMS associée. Toutefois, si vous supprimez une clé KMS, tous les alias associés à cette clé KMS sont supprimés.

Si vous spécifiez un alias comme ressource dans une politique IAM, la politique fait référence à l'alias et non à la clé KMS associée.

Chaque alias a deux formats.

Lorsque vous créez un alias, vous spécifiez son nom. AWS KMS crée l'alias ARN pour vous.

- Un ARN d'alias est un Amazon Resource Name (ARN) qui identifie l'alias de façon unique.

```
# Alias ARN
arn:aws:kms:us-west-2:111122223333:alias/<alias-name>
```

- Un nom d'alias qui est unique dans le compte et la région. Dans l' AWS KMS API, le nom de l'alias est toujours préfixé par`alias/`. Ce préfixe est omis dans la AWS KMS console.

```
# Alias name
alias/<alias-name>
```

Les alias ne sont pas secrets

Les alias peuvent être affichés en texte clair dans les CloudTrail journaux et autres sorties. N'incluez pas d'informations confidentielles ou sensibles dans le nom de l’alias.

Chaque alias est associé à une clé KMS à la fois.

L'alias et sa clé KMS doivent se trouver dans le même compte et la même région.

Vous pouvez associer un alias à n'importe quelle clé gérée par le client dans Compte AWS la même région. Cependant, vous n'êtes pas autorisé à associer un alias à une Clé gérée par AWS.

Par exemple, cette ListAliasessortie indique que l'`test-key`alias est associé à une seule clé KMS cible, qui est représentée par la `TargetKeyId` propriété.

```
{
     "AliasName": "alias/test-key",
     "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
     "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
     "CreationDate": 1593622000.191,
     "LastUpdatedDate": 1593622000.191
}
```

Plusieurs alias peuvent être associés à la même clé KMS.

Par exemple, vous pouvez associer les alias `test-key` et `project-key` à la même clé KMS.

```
{
     "AliasName": "alias/test-key",
     "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
     "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
     "CreationDate": 1593622000.191,
     "LastUpdatedDate": 1593622000.191
},
{
     "AliasName": "alias/project-key",
     "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/project-key",
     "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
     "CreationDate": 1516435200.399,
     "LastUpdatedDate": 1516435200.399
}
```

Un alias doit être unique dans un compte et une région.

Par exemple, vous ne pouvez avoir qu'un seul alias `test-key` dans chaque compte et région. Les alias sont sensibles à la casse, mais les alias qui ne diffèrent que par leur majuscule sont très propices aux erreurs. Vous ne pouvez pas modifier un nom d'alias. Toutefois, vous pouvez supprimer l'alias et créer un alias avec le nom souhaité.

Vous pouvez créer des alias avec le même nom dans des régions différentes.

Par exemple, vous pouvez avoir un alias `finance-key` dans la région USA Est (Virginie du Nord) et un alias `finance-key` en Europe (Francfort). Chaque alias serait associé à une clé KMS dans sa région. Si votre code fait référence à un nom d'alias comme `alias/finance-key`, vous pouvez l'exécuter dans plusieurs régions. Dans chaque région, il utilise une clé KMS différente. Pour plus de détails, consultez Apprenez à utiliser des alias dans vos applications.

Vous pouvez modifier la clé KMS associée à un alias

Vous pouvez utiliser cette UpdateAliasopération pour associer un alias à une autre clé KMS. Par exemple, si l'alias `finance-key` est associé à la clé KMS `1234abcd-12ab-34cd-56ef-1234567890ab`, vous pouvez le mettre à jour afin qu'il soit associé à la clé KMS `0987dcba-09fe-87dc-65ba-ab0987654321`.

Toutefois, la clé KMS actuelle et la nouvelle clé KMS doivent être du même type (toutes deux soit symétriques, soit asymétriques, soit HMAC) et avoir la même utilisation de clé (ENCRYPT_DECRYPT or SIGN_VERIFY ou GENERATE_VERIFY_MAC). Cette restriction empêche les erreurs dans le code qui utilise des alias. Si vous devez associer un alias à un autre type de clé et que vous avez atténué les risques, vous pouvez supprimer et recréer l'alias.

Certaines clés KMS n'ont pas d'alias.

Lorsque vous créez une clé KMS dans la AWS KMS console, vous devez lui attribuer un nouvel alias. Toutefois, aucun alias n'est requis lorsque vous utilisez l'CreateKeyopération pour créer une clé KMS. Vous pouvez également utiliser l'UpdateAliasopération pour modifier la clé KMS associée à un alias et l'DeleteAliasopération pour supprimer un alias. Par conséquent, certaines clés KMS peuvent avoir plusieurs alias, tandis que d'autres peuvent n'en avoir aucun.

AWS crée des alias dans votre compte

AWS crée des alias dans votre compte pour Clés gérées par AWS. Ces alias ont des noms au format `alias/aws/<service-name>`, comme `alias/aws/s3`.

Certains AWS alias n'ont pas de clé KMS. Ces alias prédéfinis sont généralement associés à un identifiant Clé gérée par AWS lorsque vous commencez à utiliser le service.

Utiliser des alias pour identifier les clés KMS

Vous pouvez utiliser un nom d'alias ou un ARN d'alias pour identifier une clé KMS dans le cadre d'opérations cryptographiques DescribeKey, et GetPublicKey. (Si la clé KMS est dans un autre Compte AWS, vous devez utiliser son ARN de clé ou ARN d'alias.) Les alias ne sont pas des identificateurs valides pour les clés KMS dans d'autres opérations AWS KMS . Pour plus d'informations sur les identificateurs de clé valides pour chaque opération d' AWS KMS API, consultez les descriptions des `KeyId` paramètres dans la référence AWS Key Management Service d'API.

Vous ne pouvez pas utiliser un nom d'alias ou un ARN d'alias pour identifier une clé KMS dans une politique IAM. Pour contrôler l'accès à une clé KMS en fonction de ses alias, utilisez les clés

de ResourceAliases condition kms : RequestAlias ou kms :. Pour plus de détails, consultez ABAC pour AWS KMS.

Contrôle de l'accès aux alias

Lorsque vous créez ou modifiez un alias, vous affectez l'alias et sa clé KMS associée. Par conséquent, les principaux qui gèrent les alias doivent avoir l'autorisation d'appeler l'opération d'alias sur l'alias et sur toutes les clés KMS affectées. Vous pouvez fournir ces autorisations à l'aide de politiques de clé, de politiques IAM et d'octrois.

> **Note**
>
> Soyez prudent lorsque vous autorisez les principaux à gérer les balises et les alias. La modification d'une balise ou d'un alias permet d'accorder ou de refuser l'autorisation d'utiliser la clé gérée par le client. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des alias pour contrôler l'accès aux clés KMS.

Pour plus d'informations sur le contrôle de l'accès à toutes les AWS KMS opérations, consultezRéférence des autorisations .

Les autorisations de création et de gestion des alias fonctionnent comme suit.

km : CreateAlias

Pour créer un alias, le principal a besoin des autorisations suivantes pour l'alias et pour la clé KMS associée.

- `kms:CreateAlias` pour l'alias. Fournissez cette autorisation dans une politique IAM attachée au principal autorisé à créer l'alias.

  L'exemple de déclaration de politique suivant spécifie un alias particulier dans l'élément `Resource`. Mais vous pouvez répertorier plusieurs alias ARNs ou spécifier un modèle d'alias, tel que « test* ». Vous pouvez également spécifier une valeur `Resource` de `"*"` pour permettre au principal de créer un alias dans le compte et la région. L'autorisation de créer un alias peut également être incluse dans une autorisation `kms:Create*` pour toutes les ressources d'un compte et d'une région.

  ```
  {
  ```

```
  "Sid": "IAMPolicyForAnAlias",
  "Effect": "Allow",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms:DeleteAlias"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/test-key"
}
```

- `kms:CreateAlias` pour la clé KMS. Cette autorisation doit être fournie dans une politique de clé ou dans une IAM politique déléguée par la politique de clé.

```
{
  "Sid": "Key policy for 1234abcd-12ab-34cd-56ef-1234567890ab",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"},
  "Action": [
    "kms:CreateAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

Vous pouvez utiliser des clés de condition pour limiter les clés KMS que vous pouvez associer à un alias. Par exemple, vous pouvez utiliser la clé de KeySpec condition kms : pour autoriser le principal à créer des alias uniquement sur des clés KMS asymétriques. Pour obtenir la liste complète des clés de condition que vous pouvez utiliser pour limiter l'autorisation `kms:CreateAlias` sur les ressources de clé KMS, veuillez consulter AWS KMS autorisations.

km : ListAliases

Pour répertorier les alias dans le compte et la région, le principal doit avoir une autorisation `kms:ListAliases` dans une politique IAM. Étant donné que cette politique n'est pas liée à une clé KMS particulière ou à une ressource d'alias, la valeur de l'élément de ressource dans la politique doit être `"*"`.

Par exemple, la déclaration de politique IAM suivante donne au principal l'autorisation de répertorier toutes les clés et tous les alias KMS dans le compte et la région.

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

km : UpdateAlias

Pour modifier la clé KMS associée à un alias, le principal a besoin de trois éléments d'autorisation : un pour l'alias, un pour la clé KMS actuelle et un pour la nouvelle clé KMS.

Par exemple, supposons que vous souhaitiez modifier l'alias `test-key` de la clé KMS avec l'ID de clé 1234abcd-12ab-34cd-56ef-1234567890ab vers la clé KMS avec l'ID de clé 0987dcba-09fe-87dc-65ba-ab0987654321. Dans ce cas, incluez des déclarations de politique similaires aux exemples de cette section.

- `kms:UpdateAlias` pour l'alias. Vous fournissez cette autorisation dans une politique IAM associée au principal. La politique IAM suivante spécifie un alias particulier. Mais vous pouvez répertorier plusieurs alias ARNs ou spécifier un modèle d'alias, tel que`"test*"`. Vous pouvez également spécifier une valeur `Resource` de `"*"` pour permettre au principal de mettre à jour un alias dans le compte et la région.

  ```
  {
    "Sid": "IAMPolicyForAnAlias",
    "Effect": "Allow",
    "Action": [
      "kms:UpdateAlias",
      "kms:ListAliases",
      "kms:ListKeys"
    ],
    "Resource": "arn:aws:kms:us-west-2:111122223333:alias/test-key"
  }
  ```

- `kms:UpdateAlias` pour la clé KMS actuellement associée à l'alias. Cette autorisation doit être fournie dans une politique de clé ou dans une IAM politique déléguée par la politique de clé.

```
{
  "Sid": "Key policy for 1234abcd-12ab-34cd-56ef-1234567890ab",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"},
  "Action": [
    "kms:UpdateAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

- `kms:UpdateAlias` pour la clé KMS que l'opération associe à l'alias. Cette autorisation doit être fournie dans une politique de clé ou dans une IAM politique déléguée par la politique de clé.

```
{
  "Sid": "Key policy for 0987dcba-09fe-87dc-65ba-ab0987654321",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"},
  "Action": [
    "kms:UpdateAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

Vous pouvez utiliser des clés de condition pour limiter l'une ou l'autre des clés KMS dans une opération `UpdateAlias`. Par exemple, vous pouvez utiliser une clé de ResourceAliases condition kms : pour autoriser le principal à mettre à jour les alias uniquement lorsque la clé KMS cible possède déjà un alias particulier. Pour obtenir la liste complète des clés de condition que vous pouvez utiliser pour limiter l'autorisation `kms:UpdateAlias` sur une ressource de clé KMS, veuillez consulter AWS KMS autorisations.

km : DeleteAlias

Pour supprimer un alias, le principal a besoin d'une autorisation pour l'alias et pour la clé KMS associée.

Comme toujours, vous devez faire preuve de prudence lorsque vous autorisez les principaux à supprimer une ressource. Toutefois, la suppression d'un alias n'a aucun effet sur la clé KMS

associée. Bien que cela puisse provoquer un échec dans une application qui s'appuie sur l'alias, si vous supprimez un alias par erreur, vous pouvez le recréer.

- `kms:DeleteAlias` pour l'alias. Fournissez cette autorisation dans une politique IAM attachée au principal autorisé à supprimer l'alias.

  L'exemple de déclaration de politique suivant spécifie un alias dans l'élément `Resource`. Mais vous pouvez répertorier plusieurs alias ARNs ou spécifier un modèle d'alias, par exemple`"test*"`, Vous pouvez également spécifier une `Resource` valeur de `"*"` pour permettre au principal de supprimer n'importe quel alias du compte et de la région.

  ```
  {
    "Sid": "IAMPolicyForAnAlias",
    "Effect": "Allow",
    "Action": [
      "kms:CreateAlias",
      "kms:UpdateAlias",
      "kms:DeleteAlias"
    ],
    "Resource": "arn:aws:kms:us-west-2:111122223333:alias/test-key"
  }
  ```

- `kms:DeleteAlias` pour la clé KMS associée. Cette autorisation doit être fournie dans une politique de clé ou dans une IAM politique déléguée par la politique de clé.

  ```
  {
    "Sid": "Key policy for 1234abcd-12ab-34cd-56ef-1234567890ab",
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"
    },
    "Action": [
      "kms:CreateAlias",
      "kms:UpdateAlias",
      "kms:DeleteAlias",
      "kms:DescribeKey"
    ],
    "Resource": "*"
  }
  ```

Limitation des autorisations d'alias

Vous pouvez utiliser des clés de condition pour limiter les autorisations d'alias lorsque la ressource est une clé KMS. Par exemple, la politique IAM suivante autorise les opérations d'alias sur les clés KMS d'un compte et d'une région en particulier. Cependant, il utilise la clé de KeyOrigin condition kms : pour limiter davantage les autorisations aux clés KMS dont le contenu clé provient de AWS KMS.

Pour obtenir la liste complète des clés de condition que vous pouvez utiliser pour limiter l'autorisation d'alias sur une ressource de clé KMS, veuillez consulter AWS KMS autorisations.

```
{
  "Sid": "IAMPolicyKeyPermissions",
  "Effect": "Allow",
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms:DeleteAlias"
  ],
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "AWS_KMS"
    }
  }
}
```

Vous ne pouvez pas utiliser de clés de condition dans une instruction de politique où la ressource est un alias. Pour limiter les alias qu'un principal peut gérer, utilisez la valeur de l'élément `Resource` de la déclaration de politique IAM qui contrôle l'accès à l'alias. Par exemple, les déclarations de politique suivantes autorisent le principal à créer, mettre à jour ou supprimer n'importe quel alias dans la région Compte AWS et, sauf si l'alias commence par`Restricted`.

```
{
  "Sid": "IAMPolicyForAnAliasAllow",
  "Effect": "Allow",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms:DeleteAlias"
  ],
```

```
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/*"
},
{
  "Sid": "IAMPolicyForAnAliasDeny",
  "Effect": "Deny",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms:DeleteAlias"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/Restricted*"
}
```

Création d'alias

Vous pouvez créer des alias dans la AWS KMS console ou à l'aide des opérations d' AWS KMS API.

L'alias doit être une chaîne de 1 à 256 caractères. Il peut contenir uniquement des caractères alphanumériques, des barres obliques (/), des traits de soulignement (_) et des tirets (-). Le nom d'alias d'une clé gérée par le client ne peut commencer par `alias/aws/`. Le préfixe `alias/aws/` est réservé à une Clé gérée par AWS.

Vous pouvez créer un alias pour une nouvelle clé KMS ou pour une clé KMS existante. Vous pouvez ajouter un alias afin qu'une clé KMS particulière soit utilisée dans un projet ou une application.

Vous pouvez également utiliser un AWS CloudFormation modèle pour créer un alias pour une clé KMS. Pour plus d'informations, consultez AWS::KMS::Alias dans le Guide de l'utilisateur AWS CloudFormation .

Utilisation de la AWS KMS console

Lorsque vous créez une clé KMS dans la AWS KMS console, vous devez créer un alias pour la nouvelle clé KMS. Pour créer un alias pour une clé KMS existante, utilisez l'option Aliases (Alias) sur la page de détails de la clé KMS.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.

3. Dans le volet de navigation, sélectionnez Clés gérées par le client. Vous ne pouvez pas gérer les alias pour Clés gérées par AWS ou Clés détenues par AWS.
4. Dans la table, choisissez l'alias ou l'ID d'une clé KMS. Ensuite, sur la page de détails de la clé KMS, choisissez l'onglet Aliases (Alias).

   Si une clé KMS possède plusieurs alias, la colonne Aliases (Alias) dans la table affiche un alias et un résumé d'alias, tel que (+n plus). Le choix du résumé d'alias vous mène directement à l'onglet Aliases (Alias) dans la page des détails de la clé KMS.
5. Dans l'onglet Aliases (Alias), choisissez Create alias (Créer un alias). Saisissez un nom d'alias et choisissez Create alias (Créer un alias).

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

> **Note**
>
> N'ajoutez pas le préfixe `alias/`. La console l'ajoute pour vous automatiquement. Si vous saisissez `alias/ExampleAlias`, le nom d'alias réel sera `alias/alias/ExampleAlias`.

Utilisation de l' AWS KMS API

Pour créer un alias, utilisez l'CreateAliasopération. Contrairement au processus de création de clés KMS dans la console, l'CreateKeyopération ne crée pas d'alias pour une nouvelle clé KMS.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

Vous pouvez utiliser l'opération `CreateAlias` pour créer un alias pour une nouvelle clé KMS sans alias. Vous pouvez également utiliser l'opération `CreateAlias` pour ajouter un alias à une clé KMS existante ou pour recréer un alias qui a été accidentellement supprimé.

Dans les opérations AWS KMS d'API, le nom de l'alias doit `alias/` commencer par un nom, tel que`alias/ExampleAlias`. L'alias doit être unique dans le compte et la région. Pour rechercher les noms d'alias déjà utilisés, utilisez l'ListAliasesopération. Le nom de l'alias est sensible à la casse.

Le `TargetKeyId` peut avoir n'importe quelle clé gérée par le client dans la même Région AWS. Pour identifier la clé KMS, utilisez son ID de clé ou son ARN de clé. Vous ne pouvez pas utiliser un autre alias.

L'exemple suivant crée l'alias `example-key` et l'associe à la clé KMS spécifiée. Ces exemples utilisent le AWS Command Line Interface (AWS CLI). Pour obtenir des exemples dans plusieurs langages de programmation, veuillez consulter Utilisation `CreateAlias` avec un AWS SDK ou une CLI.

```
$ aws kms create-alias \
    --alias-name alias/example-key \
    --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

`CreateAlias` ne renvoie aucune sortie. Pour voir le nouvel alias, utilisez l'opération `ListAliases`. Pour plus de détails, consultez Utilisation de l' AWS KMS API.

Trouvez le nom d'alias et l'ARN de l'alias pour une clé KMS

Les alias permettent de reconnaître facilement les clés KMS dans la AWS KMS console.
Vous pouvez afficher les alias d'une clé KMS dans la AWS KMS console ou en utilisant l'ListAliasesopération. L'DescribeKeyopération, qui renvoie les propriétés d'une clé KMS, n'inclut pas les alias.

Les procédures suivantes montrent comment afficher et identifier les alias associés à une clé KMS à l'aide de la AWS KMS console et de l' AWS KMS API. Les exemples AWS KMS d'API utilisent le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Utilisation de la AWS KMS console

La AWS KMS console affiche les alias associés à la clé KMS.

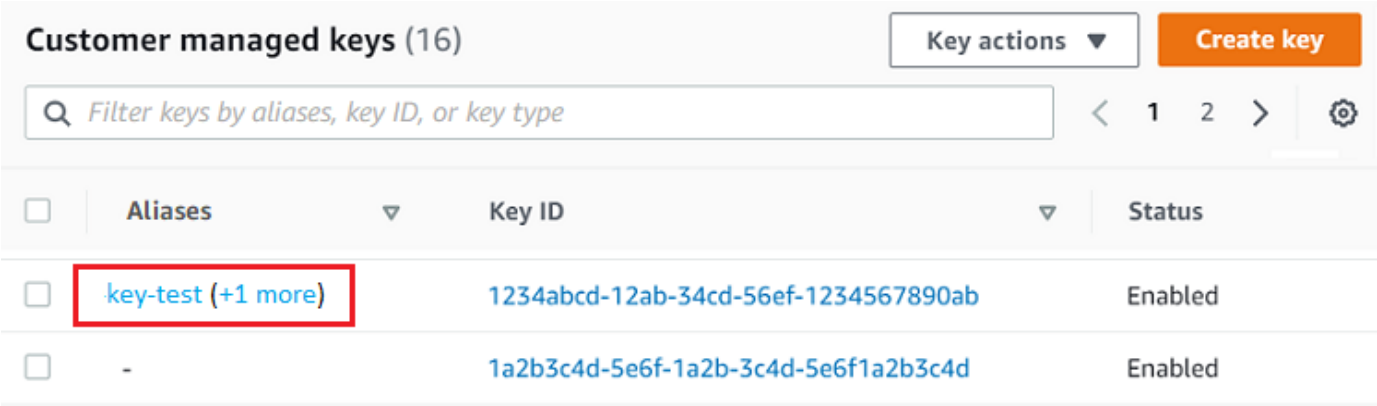
1. Ouvrez la AWS KMS console à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.

3. Pour afficher les clés de votre compte que vous créez et gérez vous-même, dans le volet de navigation, choisissez Clés gérées par le client. Pour afficher les clés de votre compte qui AWS crée et gère pour vous, dans le volet de navigation, choisissez les clés AWS gérées.
4. La colonne Aliases (Alias) affiche l'alias de chaque clé KMS. Si une clé KMS n'a pas d'alias, un tiret (-) apparaît dans la colonne Aliases (Alias).

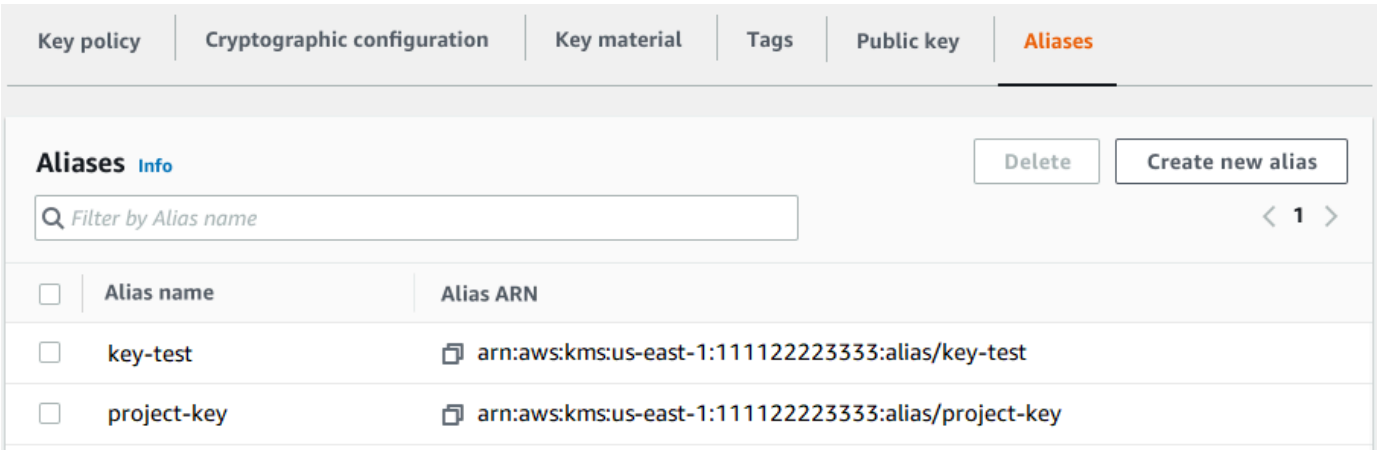
   Si une clé KMS possède plusieurs alias, la colonne Aliases (Alias) contient également un résumé d'alias, tel que (+n plus). Par exemple, la clé KMS suivante a deux alias, dont l'un est `key-test`.

   Pour trouver le nom d'alias et l'ARN d'alias de tous les alias de la clé KMS, utilisez l'onglet Aliases (Alias).

   - Pour accéder directement à l'onglet Aliases (Alias), dans la colonne Aliases (Alias), choisissez le résumé de l'alias (+n plus). Un résumé d'alias apparaît uniquement si la clé KMS comporte plusieurs alias.
   - Vous pouvez également choisir l'alias ou l'ID de clé de la clé KMS (qui ouvre la page des détails de la clé KMS), puis l'onglet Aliases (Alias). Les onglets se trouvent sous la section General configuration (Configuration générale).

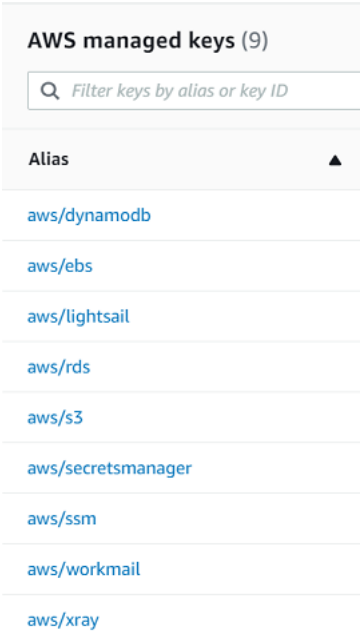


5. L'onglet Aliases (Alias) affiche le nom d'alias et l'ARN d'alias de tous les alias d'une clé KMS. Vous pouvez également créer et supprimer des alias pour la clé KMS sur cet onglet.



Clés gérées par AWS

Vous pouvez utiliser l'alias pour reconnaître un Clé gérée par AWS, comme le montre cet exemple de Clés gérées par AWSpage. Les alias de Clés gérées par AWS sont toujours au format `aws/<service-name>`. Par exemple, l'alias Clé gérée par AWS pour Amazon `aws/dynamodb` DynamoDB est.



Utilisation de l' AWS KMS API

L'ListAliasesopération renvoie le nom d'alias et l'ARN d'alias des alias du compte et de la région. La sortie inclut des alias pour Clés gérées par AWS et pour les clés gérées par le client. Les alias de Clés gérées par AWS sont toujours au format `aws/<service-name>`, comme `aws/dynamodb`.

La réponse peut également inclure des alias ne disposant pas de champ `TargetKeyId`. Il s'agit d'alias prédéfinis qui ont AWS été créés mais qui ne sont pas encore associés à une clé KMS.

```
$ aws kms list-aliases
{
    "Aliases": [
        {
            "AliasName": "alias/access-key",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/access-key",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "CreationDate": 1516435200.399,
            "LastUpdatedDate": 1516435200.399
        },
        {
            "AliasName": "alias/ECC-P521-Sign",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ECC-P521-Sign",
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1693622000.704,
            "LastUpdatedDate": 1693622000.704
        },
        {
            "AliasName": "alias/ImportedKey",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ImportedKey",
            "TargetKeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d",
            "CreationDate": 1493622000.704,
            "LastUpdatedDate": 1521097200.235
        },
        {
            "AliasName": "alias/finance-project",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/finance-project",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "CreationDate": 1604958290.014,
            "LastUpdatedDate": 1604958290.014
        },
        {
            "AliasName": "alias/aws/dynamodb",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/dynamodb",
            "TargetKeyId": "0987ab65-43cd-21ef-09ab-87654321cdef",
            "CreationDate": 1521097200.454,
            "LastUpdatedDate": 1521097200.454
        },
        {
            "AliasName": "alias/aws/ebs",
```

```
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/ebs",
            "TargetKeyId": "abcd1234-09fe-ef90-09fe-ab0987654321",
            "CreationDate": 1466518990.200,
            "LastUpdatedDate": 1466518990.200
        }
    ]
}
```

Pour obtenir tous les alias associés à une clé KMS spécifique, utilisez le paramètre `KeyId` facultatif de l'opération `ListAliases`. Le paramètre `KeyId` prend l'ID de clé ou l'ARN de clé de la clé KMS.

Cet exemple récupère tous les alias associés à la clé KMS `0987dcba-09fe-87dc-65ba-ab0987654321`.

```
$ aws kms list-aliases --key-id 0987dcba-09fe-87dc-65ba-ab0987654321
{
    "Aliases": [
        {
            "AliasName": "alias/access-key",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/access-key",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "CreationDate": "2018-01-20T15:23:10.194000-07:00",
            "LastUpdatedDate": "2018-01-20T15:23:10.194000-07:00"
        },
        {
            "AliasName": "alias/finance-project",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/finance-project",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
            "CreationDate": 1604958290.014,
            "LastUpdatedDate": 1604958290.014
        }
    ]
}
```

Le paramètre `KeyId` ne prend pas de caractères génériques, mais vous pouvez utiliser les fonctions de votre langage de programmation pour filtrer la réponse.

Par exemple, la AWS CLI commande suivante obtient uniquement les alias pour Clés gérées par AWS.

```
$ aws kms list-aliases --query 'Aliases[?starts_with(AliasName, `alias/aws/`)]'
```

Par exemple, la commande suivante obtient uniquement les alias `access-key`. Le nom de l'alias est sensible à la casse.

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/access-key`]'
[
    {
        "AliasName": "alias/access-key",
        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/access-key",
        "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "CreationDate": "2018-01-20T15:23:10.194000-07:00",
        "LastUpdatedDate": "2018-01-20T15:23:10.194000-07:00"
    }
]
```

Mettre à jour les alias

Étant donné qu'un alias est une ressource indépendante, vous pouvez modifier la clé KMS qui lui est associée. Par exemple, si l'`test-key`alias est associé à une clé KMS, vous pouvez utiliser l'UpdateAliasopération pour l'associer à une autre clé KMS. C'est l'une des nombreuses façons de faire tourner manuellement une clé KMS sans modifier ses éléments de clé. Vous pouvez également mettre à jour une clé KMS de sorte qu'une application qui utilisait une clé KMS pour de nouvelles ressources utilise désormais une clé KMS différente.

Vous ne pouvez pas mettre à jour un alias dans la AWS KMS console. En outre, vous ne pouvez pas utiliser `UpdateAlias` (ou toute autre opération) pour modifier un nom d'alias. Pour modifier un nom d'alias, supprimez l'alias actuel, puis créez un alias pour la clé KMS.

Lorsque vous mettez à jour un alias, la clé KMS actuelle et la nouvelle clé KMS doivent être du même type (toutes deux soit symétriques, soit asymétriques, soit HMAC). Elles doivent également avoir la même utilisation de la clé (`ENCRYPT_DECRYPT` ou `SIGN_VERIFY` ou GENERATE_VERIFY_MAC). Cette restriction empêche les erreurs cryptographiques dans le code qui utilise des alias.

L'exemple suivant commence par utiliser l'ListAliasesopération pour montrer que l'`test-key`alias est actuellement associé à la clé KMS`1234abcd-12ab-34cd-56ef-1234567890ab`.

```
$ aws kms list-aliases --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
    "Aliases": [
        {
            "AliasName": "alias/test-key",
```

```
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "CreationDate": 1593622000.191,
            "LastUpdatedDate": 1593622000.191
        }
    ]
}
```

Ensuite, il utilise l'opération `UpdateAlias` pour modifier la clé KMS associée à l'alias `test-key` avec la clé KMS `0987dcba-09fe-87dc-65ba-ab0987654321`. Vous n'avez pas besoin de spécifier la clé KMS actuellement associée, uniquement la nouvelle clé KMS (« cible »). Le nom de l'alias est sensible à la casse.

```
$ aws kms update-alias --alias-name 'alias/test-key' --target-key-id
 0987dcba-09fe-87dc-65ba-ab0987654321
```

Pour vérifier que l'alias est maintenant associé à la clé KMS cible, utilisez à nouveau l'opération `ListAliases`. Cette AWS CLI commande utilise le `--query` paramètre pour obtenir uniquement l'`test-key`alias. Les champs `TargetKeyId` et `LastUpdatedDate` sont mis à jour.

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/test-key`]'
[
    {
        "AliasName": "alias/test-key",
        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
        "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "CreationDate": 1593622000.191,
        "LastUpdatedDate": 1604958290.154
    }
]
```

Supprimer un alias

Vous pouvez supprimer un alias dans la AWS KMS console ou en utilisant l'DeleteAliasopération. Avant de supprimer un alias, vérifiez qu'il n'est pas en cours d'utilisation. Bien que la suppression d'un alias n'affecte pas la clé KMS associée, elle peut créer des problèmes pour toute application qui utilise l'alias. Si vous supprimez un alias par erreur, vous pouvez en créer un autre portant le même nom et l'associer à la même clé KMS ou à une clé KMS différente.

Si vous supprimez une clé KMS, tous les alias associés à cette clé KMS sont supprimés.

Utilisation de la AWS KMS console

Pour supprimer un alias dans la AWS KMS console, utilisez l'onglet Alias sur la page détaillée de la clé KMS. Vous pouvez supprimer plusieurs alias d'une clé KMS en même temps.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. Vous ne pouvez pas gérer les alias pour Clés gérées par AWS ou Clés détenues par AWS.
4. Dans la table, choisissez l'alias ou l'ID d'une clé KMS. Ensuite, sur la page de détails de la clé KMS, choisissez l'onglet Aliases (Alias).

   Si une clé KMS possède plusieurs alias, la colonne Aliases (Alias) dans la table affiche un alias et un résumé d'alias, tel que (+n plus). Le choix du résumé d'alias vous mène directement à l'onglet Aliases (Alias) dans la page des détails de la clé KMS.
5. Dans l'onglet Aliases (Alias), cochez la case en regard des alias que vous souhaitez supprimer. Ensuite, choisissez Supprimer.

Utilisation de l' AWS KMS API

Pour supprimer un alias, utilisez l'DeleteAliasopération. Cette opération supprime un alias à la fois. Le nom de l'alias est sensible à la casse et doit être précédé du préfixe `alias/`.

Par exemple, la commande suivante supprime l'alias `test-key`. Cette commande ne renvoie aucune sortie.

```
$ aws kms delete-alias --alias-name alias/test-key
```

Pour vérifier que l'alias est supprimé, utilisez l'ListAliasesopération. La commande suivante utilise le `--query` paramètre du AWS CLI pour obtenir uniquement l'`test-key`alias. Les crochets vides dans la réponse indiquent que la réponse `ListAliases` n'incluait pas d'alias `test-key`. Pour éliminer les crochets, utilisez le paramètre et la valeur `--output text`.

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/test-key`]'
[]
```

Utiliser des alias pour contrôler l'accès aux clés KMS

Vous pouvez contrôler l'accès aux clés KMS en fonction des alias associés à la clé KMS. Pour ce faire, utilisez les clés de ResourceAliases condition kms : RequestAlias et kms :. Cette fonctionnalité fait partie de la AWS KMS prise en charge du contrôle d'accès basé sur les attributs (ABAC).

La clé de condition `kms:RequestAlias` autorise ou refuse l'accès à une clé KMS en fonction de l'alias dans une requête. La clé de condition `kms:ResourceAliases` autorise ou refuse l'accès à une clé KMS en fonction des alias associés à la clé KMS.

Ces fonctionnalités ne vous permettent pas d'identifier une clé KMS à l'aide d'un alias dans l'élément `resource` d'une déclaration de politique. Lorsqu'un alias est la valeur d'un élément `resource`, la politique s'applique à la ressource d'alias et non à toute clé KMS qui pourrait lui être associée.

> **Note**
>
> Les modifications d'alias et de balise peuvent prendre jusqu'à cinq minutes pour avoir une incidence sur l'autorisation de clé KMS. Les modifications récentes peuvent être visibles dans les opérations d'API avant qu'elles n'affectent l'autorisation.

Lorsque vous utilisez des alias pour contrôler l'accès aux clés KMS, tenez compte des éléments suivants :

- Utilisez des alias pour renforcer les bonnes pratiques de l'accès le moins privilégié. N'accordez aux principaux IAM que les autorisations dont ils ont besoin pour les clés KMS qu'ils doivent utiliser ou gérer. Par exemple, utilisez des alias pour identifier les clés KMS utilisées pour un projet. Donnez ensuite à l'équipe de projet l'autorisation d'utiliser uniquement les clés KMS avec les alias du projet.
- Soyez prudent lorsque vous donnez aux principaux les autorisations `kms:CreateAlias`, `kms:UpdateAlias` ou `kms:DeleteAlias` qui leur permettent d'ajouter, de modifier et de supprimer des alias. Lorsque vous utilisez des alias pour contrôler l'accès aux clés KMS, la modification d'un alias peut donner aux principaux l'autorisation d'utiliser des clés KMS qu'ils n'avaient alors pas l'autorisation d'utiliser. Elle peut également refuser l'accès aux clés KMS dont d'autres principaux ont besoin pour réaliser leurs tâches.
- Passez en revue les principaux de votre ordinateur Compte AWS qui sont actuellement autorisés à gérer les alias et ajustez ces autorisations, si nécessaire. Les administrateurs de clés qui n'ont pas l'autorisation de modifier les politiques de clé ou de créer des octrois peuvent contrôler l'accès aux clés KMS s'ils sont autorisés à gérer les alias.

Par exemple, la console Politique de clé par défaut pour les administrateurs de clés comprend les autorisations `kms:CreateAlias`, `kms:DeleteAlias` et `kms:UpdateAlias`. Les politiques IAM peuvent donner des autorisations d'alias pour toutes les clés KMS de votre Compte AWS. Par exemple, la politique AWSKeyManagementServicePowerUsergérée permet aux principaux de créer, de supprimer et de répertorier des alias pour toutes les clés KMS, mais pas de les mettre à jour.

- Avant de définir une politique qui dépend d'un alias, passez en revue les alias figurant sur les clés KMS de votre Compte AWS. Assurez-vous que votre politique s'applique uniquement aux alias que vous avez l'intention d'inclure. Utilisez CloudTrail les journaux et les CloudWatch alarmes pour vous avertir des modifications d'alias susceptibles d'affecter l'accès à vos clés KMS. La ListAliasesréponse inclut également la date de création et la date de dernière mise à jour pour chaque alias.
- Les conditions de politique d'alias utilisent la correspondance de modèles ; elles ne sont pas liées à une instance particulière d'un alias. Une politique qui utilise des clés de condition basées sur des alias affecte tous les alias nouveaux et existants qui correspondent au modèle. Si vous supprimez et recréez un alias qui correspond à une condition de politique, la condition s'applique au nouvel alias, comme c'était le cas pour l'ancien.

La clé de condition `kms:RequestAlias` repose sur l'alias spécifié explicitement dans une demande d'opération. La clé de condition `kms:ResourceAliases` dépend des alias associés à une clé KMS, même s'ils n'apparaissent pas dans la demande.

km : RequestAlias

Autoriser ou refuser l'accès à une clé KMS en fonction de l'alias qui identifie la clé KMS dans une demande. Vous pouvez utiliser la clé de RequestAlias condition kms : dans une politique clé ou une politique IAM. Elle s'applique aux opérations qui utilisent un alias pour identifier une clé KMS dans une demande, à savoir les opérations cryptographiques DescribeKey, et GetPublicKey. Il n'est pas valide pour les opérations d'alias, telles que CreateAliasou DeleteAlias.

Dans la clé de condition, spécifiez un Nom d'alias ou un modèle de nom d'alias. Vous ne pouvez pas spécifier d'ARN d'alias.

Par exemple, la déclaration de politique de clé suivante autorise les principaux à utiliser les opérations spécifiées sur la clé KMS. L'autorisation est en vigueur uniquement lorsque la demande utilise un alias qui inclut `alpha` pour identifier la clé KMS.

```
{
  "Sid": "Key policy using a request alias condition",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/alpha-developer"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:RequestAlias": "alias/*alpha*"
    }
  }
}
```

L'exemple suivant de demande d'un principal autorisé remplirait la condition. Cependant, une demande qui a utilisé un ID de clé, un ARN de clé ou un alias différent ne remplirait pas la condition, même si ces valeurs identifiaient la même clé KMS.

```
$ aws kms describe-key --key-id "arn:aws:kms:us-west-2:111122223333:alias/project-
alpha"
```

km : ResourceAliases

Autorisez ou refusez l'accès à une clé KMS en fonction des alias associés à la clé KMS, même si l'alias n'est pas utilisé dans une demande. La clé de ResourceAliases condition kms : vous permet de spécifier un alias ou un modèle d'alias, par exemple`alias/test*`, afin que vous puissiez l'utiliser dans une politique IAM pour contrôler l'accès à plusieurs clés KMS dans la même région. Il est valide pour toute AWS KMS opération utilisant une clé KMS.

Par exemple, la politique IAM suivante permet aux principaux d'appeler en deux les opérations spécifiées sur les clés KMS. Comptes AWS Toutefois, l'autorisation s'applique uniquement aux clés KMS associées aux alias commençant par `restricted`.

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

```
  "Statement": [
    {
      "Sid": "AliasBasedIAMPolicy",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": [
        "arn:aws:kms:*:111122223333:key/*",
        "arn:aws:kms:*:444455556666:key/*"
      ],
      "Condition": {
        "ForAnyValue:StringLike": {
          "kms:ResourceAliases": "alias/restricted*"
        }
      }
    }
  ]
}
```

La condition `kms:ResourceAliases` est une condition de la ressource, pas la demande. Dès lors, une demande qui ne spécifie pas l'alias peut toujours satisfaire la condition.

L'exemple de demande suivant, qui spécifie un alias correspondant, satisfait à la condition.

```
$ aws kms enable-key-rotation --key-id "alias/restricted-project"
```

Toutefois, l'exemple de demande suivant satisfait également la condition, à condition que la clé KMS spécifiée ait un alias qui commence par `restricted`, même si cet alias n'est pas utilisé dans la demande.

```
$ aws kms enable-key-rotation --key-id "1234abcd-12ab-34cd-56ef-1234567890ab"
```

Apprenez à utiliser des alias dans vos applications

Vous pouvez utiliser un alias pour représenter une clé KMS dans votre code d'application. `KeyId`Paramètre utilisé dans les opérations AWS KMS cryptographiques DescribeKey, et GetPublicKeyaccepte un nom d'alias ou un ARN d'alias.

Par exemple, la commande `GenerateDataKey` suivante utilise un nom d'alias (`alias/finance`) pour identifier une clé KMS. Le nom de l'alias est la valeur du paramètre `KeyId`.

```
$ aws kms generate-data-key --key-id alias/finance --key-spec AES_256
```

Si la clé KMS se trouve dans un autre Compte AWS, vous devez utiliser un ARN de clé ou un alias ARN dans ces opérations. Lorsque vous utilisez un ARN d'alias, n'oubliez pas que l'alias d'une clé KMS est défini dans le compte propriétaire de la clé KMS et peut différer d'une région à l'autre. Pour rechercher l'ARN d'alias, veuillez consulter Trouvez le nom d'alias et l'ARN de l'alias pour une clé KMS.

Par exemple, la commande `GenerateDataKey` suivante utilise une clé KMS qui ne se trouve pas dans le compte de l'appelant. L'alias `ExampleAlias` est associé à la clé KMS dans le compte et la région spécifiés.

```
$ aws kms generate-data-key --key-id arn:aws:kms:us-west-2:444455556666:alias/
ExampleAlias --key-spec AES_256
```

L'une des utilisations les plus performantes des alias est au niveau des applications qui s'exécutent dans plusieurs Régions AWS. Par exemple, vous utilisez peut-être une application mondiale qui utilise une Clé KMS asymétriques RSA pour la signature et la vérification.

- Dans la région USA Ouest (Oregon) (us-west-2), vous souhaitez utiliser `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`.
- En Europe (Francfort) (eu-central-1), vous souhaitez utiliser `arn:aws:kms:eu-central-1:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321`.
- Dans la région Asie-Pacifique (Singapour) (ap-southeast-1), vous souhaitez utiliser `arn:aws:kms:ap-southeast-1:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d`.

Vous pouvez créer une version différente de votre application dans chaque région ou utiliser un dictionnaire ou une instruction switch pour sélectionner la clé KMS appropriée pour chaque région. Toutefois, il est beaucoup plus facile de créer un alias avec le même nom d'alias dans chaque région. Rappelez-vous que ne nom de l'alias est sensible à la casse.

```
aws --region us-west-2 kms create-alias \
    --alias-name alias/new-app \
```

```
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab

aws --region eu-central-1 kms create-alias \
    --alias-name alias/new-app \
    --key-id arn:aws:kms:eu-central-1:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321

aws --region ap-southeast-1 kms create-alias \
    --alias-name alias/new-app \
    --key-id arn:aws:kms:ap-
southeast-1:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d
```

Ensuite, utilisez l'alias dans votre code. Lorsque votre code s'exécute dans chaque région, l'alias fait référence à sa clé KMS associée dans cette région. Par exemple, ce code appelle l'opération Sign avec un nom d'alias.

```
aws kms sign --key-id alias/new-app \
    --message $message \
    --message-type RAW \
    --signing-algorithm RSASSA_PSS_SHA_384
```

Toutefois, il existe un risque que l'alias soit supprimé ou mis à jour pour être associé à une autre clé KMS. Dans ce cas, les tentatives de l'application pour vérifier les signatures à l'aide du nom d'alias échouent et vous devrez peut-être recréer ou mettre à jour l'alias.

Pour atténuer ce risque, soyez prudent lorsque vous autorisez les principaux à gérer les alias que vous utilisez dans votre application. Pour plus de détails, consultez Contrôle de l'accès aux alias.

Il existe plusieurs autres solutions pour les applications qui chiffrent les données dans plusieurs Régions AWS, y compris le AWS Encryption SDK.

Rechercher des alias dans les journaux AWS CloudTrail

Vous pouvez utiliser un alias pour représenter un AWS KMS key dans une opération d' AWS KMS API. Lorsque vous le faites, l'alias et l'ARN de la clé KMS sont enregistrés dans l'entrée du AWS CloudTrail journal de l'événement. L'alias apparaît dans le champ `requestParameters`. L'ARN de clé apparaît dans le champ `resources`. Cela est vrai même lorsqu'un AWS service utilise un Clé gérée par AWS dans votre compte.

Par exemple, la GenerateDataKeydemande suivante utilise l'`project-key`alias pour représenter une clé KMS.

```
$ aws kms generate-data-key --key-id alias/project-key --key-spec AES_256
```

Lorsque cette demande est enregistrée dans le CloudTrail journal, l'entrée du journal inclut à la fois l'alias et l'ARN de la clé KMS réellement utilisée.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDE",
        "arn": "arn:aws:iam::111122223333:role/ProjectDev",
        "accountId": "111122223333",
        "accessKeyId": "FFHIJ",
        "userName": "example-dev"
    },
    "eventTime": "2020-06-29T23:36:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.205.123.000",
    "userAgent": "aws-cli/1.18.89 Python/3.6.10
 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.12",
    "requestParameters": {
        "keyId": "alias/project-key",
        "keySpec": "AES_256"
    },
    "responseElements": null,
    "requestID": "d93f57f5-d4c5-4bab-8139-5a1f7824a363",
    "eventID": "d63001e2-dbc6-4aae-90cb-e5370aca7125",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
```

```
}
```

Pour plus de détails sur les AWS KMS opérations de journalisation dans CloudTrail les journaux, consultezJournalisation des appels d' AWS KMS API avec AWS CloudTrail.

Tags dans AWS KMS

Une balise est une étiquette de métadonnées facultative que vous pouvez attribuer (ou AWS attribuer) à une AWS ressource. Chaque balise est constituée d'une clé de balise et d'un valeur de balise, qui sont toutes deux des chaînes sensibles à la casse. La valeur de balise peut être une chaîne vide (nulle). Chaque balise d'une ressource doit avoir une clé de balise différente, mais vous pouvez ajouter la même balise à plusieurs AWS ressources. Chaque ressource peut avoir jusqu'à 50 balises créées par l'utilisateur.

N'incluez pas d'informations confidentielles ou sensibles dans la clé de balise ou la valeur de balise. Les tags sont accessibles à de nombreuses personnes Services AWS, y compris pour la facturation.

Dans AWS KMS, vous pouvez ajouter des balises à une clé gérée par le client lorsque vous créez la clé KMS, et baliser ou débaliser les clés KMS existantes, sauf si elles sont en attente de suppression. Vous ne pouvez pas étiqueter les alias Clés gérées par AWS, les banques de clés personnalisées ou Clés détenues par AWS les clés KMS dans d'autres Comptes AWS. Les balises sont facultatives, mais peuvent être très utiles.

Par exemple, vous pouvez ajouter une balise `"Project"="Alpha"` à toutes les clés KMS et compartiments Amazon S3 que vous utilisez pour le projet Alpha.

```
TagKey   = "Project"
TagValue = "Alpha"
```

Pour obtenir des informations générales sur les balises, notamment leur format et leur syntaxe, consultez la section AWS Ressources de balisage dans le Référence générale d'Amazon Web Services.

Les balises vous permettent d’effectuer les actions suivantes :

- Identifiez et organisez vos AWS ressources. De nombreux AWS services prennent en charge le balisage. Vous pouvez donc attribuer le même tag aux ressources de différents services pour indiquer que les ressources sont liées. Par exemple, vous pouvez attribuer la même balise à une clé KMS et à un volume ou à un secret Amazon Elastic Block Store (Amazon EBS). AWS Secrets Manager Vous pouvez également utiliser des balises pour identifier les clés KMS pour l'automatisation.
- Suivez vos AWS coûts. Lorsque vous ajoutez des balises à vos AWS ressources, AWS génère un rapport de répartition des coûts avec l'utilisation et les coûts agrégés par balises. Vous pouvez

utiliser cette fonctionnalité pour suivre AWS KMS les coûts d'un projet, d'une application ou d'un centre de coûts.

Pour en savoir plus sur l'utilisation des balises pour la répartition des coûts, veuillez consulter Utilisation des balises de répartition des coûts dans le Guide de l'utilisateur AWS Billing . Pour obtenir des informations sur les règles des clés et valeurs de balise, veuillez consulter Restrictions encadrant les balises définies par l'utilisateur dans le Guide de l'utilisateur AWS Billing .

- Contrôlez l'accès à vos AWS ressources. L'autorisation et le refus d'accès aux clés KMS en fonction de leurs balises font partie de la AWS KMS prise en charge du contrôle d'accès basé sur les attributs (ABAC). Pour plus d'informations sur le contrôle de l'accès en AWS KMS keys fonction de leurs balises, consultezUtiliser des balises pour contrôler l'accès aux clés KMS. Pour des informations plus générales sur l'utilisation de balises pour contrôler l'accès aux AWS ressources, consultez la section Contrôle de l'accès aux AWS ressources à l'aide de balises de ressources dans le guide de l'utilisateur IAM.

AWS KMS écrit une entrée dans votre AWS CloudTrail journal lorsque vous utilisez les ListResourceTagsopérations TagResourceUntagResource, ou.

Rubriques

Contrôle de l'accès aux balises

Pour ajouter, afficher et supprimer des balises, que ce soit dans la AWS KMS console ou à l'aide de l'API, les directeurs doivent disposer d'autorisations de balisage. Vous pouvez fournir ces autorisations dans les politiques de clé. Vous pouvez également les fournir dans les politiques IAM (y compris les politiques de point de terminaison d'un VPC), mais seulement si la politique de clé le permet. La politique AWSKeyManagementServicePowerUsergérée permet aux principaux d'étiqueter, de débaliser et de répertorier les balises sur toutes les clés KMS auxquelles le compte peut accéder.

Vous pouvez également limiter ces autorisations en utilisant des clés de condition AWS globales pour les balises. Dans AWS KMS, ces conditions peuvent contrôler l'accès aux opérations de marquage, telles que TagResourceet UntagResource.

> **Note**
>
> Soyez prudent lorsque vous autorisez les principaux à gérer les balises et les alias. La modification d'une balise ou d'un alias permet d'accorder ou de refuser l'autorisation d'utiliser la clé gérée par le client. Pour plus de détails, veuillez consulter ABAC pour AWS KMS et Utiliser des balises pour contrôler l'accès aux clés KMS.

Pour obtenir des exemples de politiques et plus d'informations, veuillez consulter Contrôle de l'accès en fonction des clés de balises dans le Guide de l'utilisateur IAM.

Les autorisations de création et de gestion de balises fonctionnent comme suit.

km : TagResource

Autorise les principaux à ajouter ou à modifier des balises. Pour ajouter des balises lors de la création d'une clé KMS, le principal doit disposer d'une autorisation dans une politique IAM qui n'est pas limitée à des clés KMS particulières.

km : ListResourceTags

Permet aux principaux d'afficher les balises sur les clés KMS.

km : UntagResource

Permet aux principaux de supprimer des balises des clés KMS.

Autorisations de balises dans les politiques

Vous pouvez fournir des autorisations d'étiquetage dans une politique de clé ou une politique IAM. Par exemple, l'exemple de politique de clé suivant donne à certains utilisateurs l'autorisation d'étiqueter la clé KMS. Il accorde à tous les utilisateurs qui peuvent endosser les rôles Administrateur ou Développeur d'exemple l'autorisation d'afficher les balises.

```
{
  "Version": "2012-10-17",
  "Id": "example-key-policy",
```

```
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow all tagging permissions",
      "Effect": "Allow",
      "Principal": {"AWS": [
        "arn:aws:iam::111122223333:user/LeadAdmin",
        "arn:aws:iam::111122223333:user/SupportLead"
      ]},
      "Action": [
          "kms:TagResource",
          "kms:ListResourceTags",
          "kms:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow roles to view tags",
      "Effect": "Allow",
      "Principal": {"AWS": [
        "arn:aws:iam::111122223333:role/Administrator",
        "arn:aws:iam::111122223333:role/Developer"
      ]},
      "Action": "kms:ListResourceTags",
      "Resource": "*"
    }
  ]
}
```

Pour accorder aux principaux l'autorisation d'étiquetage sur plusieurs clés KMS, vous pouvez utiliser une politique IAM. Pour que cette politique soit efficace, la politique de clé pour chaque clé KMS doit autoriser le compte à utiliser des politiques IAM pour contrôler l'accès à la clé KMS.

Par exemple, la politique IAM suivante permet aux principaux de créer des clés KMS. Il leur permet également de créer et de gérer des balises sur toutes les clés KMS du compte spécifié. Cette combinaison permet aux principaux d'utiliser le paramètre Tags de l'CreateKeyopération pour ajouter des balises à une clé KMS lors de sa création.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKeys",
      "Effect": "Allow",
      "Action": "kms:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyTags",
      "Effect": "Allow",
      "Action": [
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ListResourceTags"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    }
  ]
}
```

Limitation des autorisations de balises

Vous pouvez limiter les autorisations d'étiquetage en utilisant des conditions de politique. Les conditions de politique suivantes peuvent être appliquées aux autorisations `kms:TagResource` et `kms:UntagResource`. Par exemple, vous pouvez utiliser la condition `aws:RequestTag/`*`tag-key`* pour permettre à un principal d'ajouter uniquement des balises particulières, ou empêcher un principal d'ajouter des balises avec des clés de balise particulières. Sinon, vous pouvez utiliser la condition `kms:KeyOrigin` pour empêcher les principaux d'étiqueter ou de désétiqueter les clés KMS avec des éléments de clé importés.

- lois : RequestTag
- aws :ResourceTag/tag-key (politiques IAM uniquement)
- lois : TagKeys
- km : CallerAccount
- km : KeySpec
- km : KeyUsage
- km : KeyOrigin

- km : ViaService

La bonne pratique à adopter lorsque vous utilisez des balises pour contrôler l'accès aux clés KMS consiste à utiliser la clé de condition `aws:RequestTag/`*`tag-key`* ou `aws:TagKeys` pour déterminer quelles balises (ou clés de balise) sont autorisées.

Par exemple, la politique IAM suivante est similaire à la précédente. Toutefois, cette politique permet aux principaux de créer des balises (`TagResource`) et de supprimer des balises `UntagResource` uniquement pour les balises avec une clé de balise `Project`.

Étant donné que `TagResource` les `UntagResource` demandes peuvent inclure plusieurs balises, vous devez spécifier un opérateur `ForAllValues` ou un `ForAnyValue` ensemble avec la TagKeys condition aws :. L'opérateur `ForAnyValue` exige qu'au moins l'une des clés de balise dans la demande corresponde à l'une des clés de balise dans la politique. L'opérateur `ForAllValues` exige que toutes les clés de balise dans la demande correspondent à l'une des clés de balise dans la politique. L'`ForAllValues`opérateur renvoie également `true` si la demande ne contient aucune balise, mais TagResource UntagResource échoue si aucune balise n'est spécifiée. Pour plus de détails sur les opérateurs d'ensemble, veuillez consulter Utiliser plusieurs clés et valeurs dans le Guide de l'utilisateur IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKey",
      "Effect": "Allow",
      "Action": "kms:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyViewAllTags",
      "Effect": "Allow",
      "Action": "kms:ListResourceTags",
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    },
    {
      "Sid": "IAMPolicyManageTags",
      "Effect": "Allow",
      "Action": [
        "kms:TagResource",
        "kms:UntagResource"
```

```
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
          "ForAllValues:StringEquals": {"aws:TagKeys": "Project"}
      }
    }
  ]
}
```

Ajouter des balises à une clé KMS

Les balises permettent d'identifier et d'organiser vos AWS ressources. Vous pouvez ajouter des balises à une clé gérée par le client lorsque vous créez la clé KMS, ou ajouter des balises aux clés KMS existantes. Vous ne pouvez pas étiqueter Clés gérées par AWS.

Les procédures suivantes montrent comment ajouter des balises aux clés gérées par le client à l'aide de la AWS KMS console et de AWS KMS l'API. Les exemples AWS KMS d'API utilisent le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Rubriques

- Ajouter des balises lors de la création d'une clé KMS
- Ajouter des balises aux clés KMS existantes

Ajouter des balises lors de la création d'une clé KMS

Vous pouvez ajouter des balises à une clé KMS lorsque vous créez la clé à l'aide de la AWS KMS console ou de l'CreateKeyopération. Pour ajouter des balises lors de la création d'une clé KMS, vous devez disposer d'une `kms:TagResource` autorisation dans une politique IAM en plus des autorisations requises pour créer des clés KMS. Au minimum, l'autorisation doit couvrir toutes les clés KMS du compte et de la région. Pour plus de détails, consultez Contrôle de l'accès aux balises.

Utilisation de la AWS KMS console

Pour ajouter des balises lors de la création d'une clé KMS dans la console, vous devez disposer des autorisations requises pour afficher les clés KMS dans la console, en plus des autorisations requises pour étiqueter et créer des clés KMS. Au minimum, l'autorisation doit couvrir toutes les clés KMS du compte et de la région.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas gérer les balises d'une Clé gérée par AWS.)
4. Choisissez le type de clé, puis choisissez Next (Suivant).
5. Saisissez un alias et une description facultative.
6. Saisissez une clé de balise et une valeur de balise facultative. Pour ajouter des balises supplémentaires, sélectionnez Add tag (Ajouter une balise). Pour supprimer une balise, choisissez Remove (Supprimer). Lorsque vous avez terminé d'étiqueter votre nouvelle clé KMS, cliquez sur Next (Suivant).
7. Terminez la création de votre clé KMS.

Utilisation de l' AWS KMS API

Pour spécifier des balises lors de la création de clés à l'aide de l'CreateKeyopération, utilisez le `Tags` paramètre de l'opération.

La valeur du paramètre `Tags` de `CreateKey` est une collection de paires de clés et de valeurs de balises sensibles à la casse. Chaque balise d'une clé KMS doit avoir un nom de balise différent. La valeur de balise peut être une chaîne vide ou nulle.

Par exemple, la AWS CLI commande suivante crée une clé KMS de chiffrement symétrique avec une `Project:Alpha` balise. Lorsque vous spécifiez plusieurs paires clé-valeur, utilisez un espace pour séparer chaque paire.

```
$ aws kms create-key --tags TagKey=Project,TagValue=Alpha
```

Lorsque cette commande aboutit, elle renvoie un objet `KeyMetadata` contenant des informations sur la nouvelle clé KMS. Cependant, l'objet `KeyMetadata` n'inclut pas les balises. Pour obtenir les balises, utilisez l'ListResourceTagsopération.

Ajouter des balises aux clés KMS existantes

Vous pouvez ajouter des balises à vos clés KMS existantes gérées par le client dans la AWS KMS console ou en utilisant l'TagResourceopération. Pour ajouter des balises, vous devez disposer d'une

autorisation de balisage sur la clé KMS. Vous pouvez obtenir cette autorisation à partir de la politique de clé pour la clé KMS ou, si la politique de clé l'autorise, à partir d'une politique IAM qui inclut la clé KMS.

Utilisation de la AWS KMS console

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas gérer les balises d'une Clé gérée par AWS.)
4. Vous pouvez utiliser le filtre du tableau pour afficher uniquement les clés KMS avec des balises particulières. Pour plus de détails, voir Afficher les balises à l'aide de la AWS KMS console.
5. Sélectionnez la case à cocher en regard de l'alias d'une clé KMS.
6. Sélectionnez Actions de clé, Ajouter ou modifier des balises.
7. Sur la page de détails de la clé KMS, sélectionnez l'onglet Tags (Balises).
   - Pour créer votre première balise, sélectionnez Create tag (Créer une balise), saisissez une clé et une valeur de balise (requis), puis sélectionnez Save (Enregistrer).

     Si vous laissez la valeur de balise vide, la valeur de balise réelle est une chaîne nulle ou vide.
   - Pour ajouter une balise, sélectionnez Edit (Modifier), choisissez Add tag (Ajouter une balise), saisissez une clé et une valeur de balise, puis choisissez Save (Enregistrer).
8. Sélectionnez Enregistrer pour enregistrer les modifications.

Utilisation de l' AWS KMS API

L'TagResourceopération ajoute une ou plusieurs balises à une clé KMS. Vous ne pouvez pas utiliser cette opération pour ajouter des balises dans un autre Compte AWS. Vous pouvez également utiliser cette TagResource opération pour modifier des balises existantes. Pour de plus amples informations, veuillez consulter the section called “Modifier les balises”.

Pour ajouter une balise, spécifiez de nouvelles clé et valeur de balises. Chaque balise d'une clé KMS doit avoir une clé de balise différente. La valeur de balise peut être une chaîne vide ou nulle.

Par exemple, la commande suivante ajoute les balises **`Purpose`** et **`Department`** à un exemple de clé KMS.

```
$ aws kms tag-resource \
        --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
        --tags TagKey=Purpose,TagValue=Pretest TagKey=Department,TagValue=Finance
```

Lorsque cette commande aboutit, elle ne renvoie pas de sortie. Pour afficher les balises d'une clé KMS, utilisez l'ListResourceTagsopération.

Modifier les balises associées à une clé KMS

Les balises permettent d'identifier et d'organiser vos AWS ressources. Vous pouvez modifier les balises associées aux clés KMS gérées par le client dans la AWS KMS console ou en utilisant cette TagResourceopération. Vous ne pouvez pas modifier les balises d'un Clé gérée par AWS.

Les procédures suivantes montrent comment modifier les balises associées à une clé KMS. Les exemples AWS KMS d'API utilisent le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Utilisation de la AWS KMS console

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas modifier les balises d'un Clé gérée par AWS)
4. Vous pouvez utiliser le filtre du tableau pour afficher uniquement les clés KMS avec des balises particulières. Pour plus de détails, voir Afficher les balises à l'aide de la AWS KMS console.
5. Sélectionnez la case à cocher en regard de l'alias d'une clé KMS.
6. Sélectionnez Actions de clé, Ajouter ou modifier des balises.
7. Sur la page de détails de la clé KMS, sélectionnez l'onglet Tags (Balises).
   - Pour modifier le nom ou la valeur d'une balise, choisissez Modifier, effectuez les modifications nécessaires, puis choisissez Enregistrer.
8. Sélectionnez Enregistrer pour enregistrer les modifications.

Utilisation de l' AWS KMS API

L'TagResourceopération ajoute une ou plusieurs balises à une clé gérée par le client ; Cependant, vous pouvez également utiliser TagResourcepour modifier la valeur d'une balise existante. Vous ne pouvez pas utiliser cette opération pour ajouter ou modifier des balises dans un autre Compte AWS.

Pour modifier une balise, spécifiez une clé de balise existante et une nouvelle valeur de balise. Chaque balise d'une clé KMS doit avoir une clé de balise différente. La valeur de balise peut être une chaîne vide ou nulle.

Par exemple, cette commande modifie la valeur de la balise `Purpose` de `Pretest` en `Test`.

```
$ aws kms tag-resource \
        --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
        --tags TagKey=Purpose,TagValue=Test
```

Supprimer les balises associées à une clé KMS

Les balises permettent d'identifier et d'organiser vos AWS ressources. Vous pouvez supprimer les balises associées aux clés KMS gérées par le client dans la AWS KMS console ou en utilisant cette UntagResourceopération. Vous ne pouvez pas modifier ou supprimer les balises d'un Clé gérée par AWS.

Les procédures suivantes montrent comment supprimer des balises d'une clé KMS. Les exemples AWS KMS d'API utilisent le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Utilisation de la AWS KMS console

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas gérer les balises d'une Clé gérée par AWS.)
4. Vous pouvez utiliser le filtre du tableau pour afficher uniquement les clés KMS avec des balises particulières. Pour plus de détails, voir Afficher les balises à l'aide de la AWS KMS console.
5. Sélectionnez la case à cocher en regard de l'alias d'une clé KMS.

6. Sélectionnez Actions de clé, Ajouter ou modifier des balises.
7. Sur la page de détails de la clé KMS, sélectionnez l'onglet Tags (Balises).

   - Pour supprimer une balise, choisissez Modifier. Sur la ligne de la balise, choisissez Supprimer, puis choisissez Enregistrer.
8. Sélectionnez Enregistrer pour enregistrer les modifications.

Utilisation de l' AWS KMS API

L'UntagResourceopération supprime les balises d'une clé KMS. Pour identifier les balises à supprimer, spécifiez les clés de balise. Vous ne pouvez pas utiliser cette opération pour supprimer des balises des clés KMS dans un autre Compte AWS.

Lorsque l'opération `UntagResource` réussit, elle ne renvoie aucune sortie. En outre, si la clé de balise spécifiée n'est pas trouvée sur la clé KMS, elle ne génère pas d'exception ou ne renvoie pas de réponse. Pour vérifier que l'opération a fonctionné, ListResourceTagsutilisez-la.

Par exemple, cette commande supprime la balise **`Purpose`** et sa valeur de la clé KMS spécifiée.

```
$ aws kms untag-resource --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --tag-keys
 Purpose
```

Afficher les balises associées à une clé KMS

Les balises permettent d'identifier et d'organiser vos AWS ressources. Vous pouvez afficher les balises associées aux clés KMS gérées par le client dans la AWS KMS console ou en utilisant l'ListResourceTagsopération.

Les procédures suivantes montrent comment trouver les balises associées à une clé KMS spécifique. Les exemples AWS KMS d'API utilisent le AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Utilisation de la AWS KMS console

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.

3. Dans le volet de navigation, sélectionnez Clés gérées par le client. (Vous ne pouvez pas gérer les balises d'une Clé gérée par AWS.)
4. Vous pouvez utiliser le filtre du tableau pour afficher uniquement les clés KMS avec des balises particulières.

   Pour afficher uniquement les clés KMS avec une balise particulière, choisissez la zone de filtre, choisissez la clé de balise, puis choisissez parmi les valeurs réelles de balise. Vous pouvez également saisir l'ensemble ou une partie de la valeur de balise.

   Le tableau résultant affiche toutes les clés KMS avec la balise choisie. Cependant, il n'affiche pas la balise. Pour afficher la balise, choisissez l'ID de clé ou l'alias de la clé KMS et, sur sa page de détails, choisissez l'option Tags (Balises). Les onglets apparaissent sous la section General configuration (Configuration générale).

   Ce filtre nécessite à la fois la clé de balise et la valeur de balise. Il ne trouvera pas de clés KMS en tapant uniquement la clé de balise ou seulement sa valeur. Pour filtrer les balises en fonction de la totalité ou d'une partie de la clé ou de la valeur de balise, utilisez l'ListResourceTagsopération pour obtenir les clés KMS balisées, puis utilisez les fonctionnalités de filtrage de votre langage de programmation.
5. Sélectionnez la case à cocher en regard de l'alias d'une clé KMS.
6. Sélectionnez Actions de clé, Ajouter ou modifier des balises.
7. Sur la page de détails de la clé KMS, sélectionnez l'onglet Tags (Balises).

Utilisation de l' AWS KMS API

L'ListResourceTagsopération obtient les balises d'une clé KMS. Le paramètre `KeyId` est obligatoire. Vous ne pouvez pas utiliser cette opération pour afficher les balises sur les clés KMS dans un autre Compte AWS.

Par exemple, la commande suivante obtient les balises pour un exemple de clé KMS.

```
$ aws kms list-resource-tags --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

  "Truncated": false,
  "Tags": [
      {
        "TagKey": "Project",
        "TagValue": "Alpha"
```

```
    },
    {
      "TagKey": "Purpose",
      "TagValue": "Test"
    },
    {
      "TagKey": "Department",
      "TagValue": "Finance"
    }
  ]
}
```

Utiliser des balises pour contrôler l'accès aux clés KMS

Vous pouvez contrôler l'accès à AWS KMS keys en fonction des balises figurant sur la clé KMS. Par exemple, vous pouvez écrire une politique IAM qui permet aux principaux d'activer et de désactiver uniquement les clés KMS possédant une balise particulière. Vous pouvez également utiliser une politique IAM pour empêcher les principaux d'utiliser des clés KMS dans les opérations de chiffrement, sauf si la clé KMS possède une balise particulière.

Cette fonctionnalité fait partie de la AWS KMS prise en charge du contrôle d'accès basé sur les attributs (ABAC). Pour plus d'informations sur l'utilisation de balises pour contrôler l'accès aux AWS ressources, voir À quoi sert ABAC ? AWS et le contrôle de l'accès aux AWS ressources à l'aide de balises de ressources dans le guide de l'utilisateur IAM. Pour obtenir de l'aide pour résoudre les problèmes d'accès liés à l'ABAC, veuillez consulter Résolution des problèmes liés à ABAC pour AWS KMS.

> **Note**
>
> Les modifications d'alias et de balises peuvent prendre jusqu'à cinq minutes pour affecter l'autorisation de clé KMS. Les modifications récentes peuvent être visibles dans les opérations d'API avant qu'elles n'affectent l'autorisation.

AWS KMS prend en charge la clé contextuelle de condition globale aws :ResourceTag/tag-key, qui vous permet de contrôler l'accès aux clés KMS en fonction des balises figurant sur la clé KMS. Étant donné que plusieurs clés KMS peuvent avoir la même balise, cette fonction vous permet d'appliquer l'autorisation à un ensemble sélectionné de clés KMS. Vous pouvez également facilement modifier les clés KMS dans l'ensemble en changeant leurs balises.

Dans AWS KMS, la clé de `aws:ResourceTag/tag-key` condition n'est prise en charge que dans les politiques IAM. Il n'est pas pris en charge dans les politiques clés, qui ne s'appliquent qu'à une seule clé KMS, ni dans les opérations qui n'utilisent pas une clé KMS particulière, telles que les ListAliasesopérations ListKeysor.

Le contrôle de l'accès à l'aide de balises offre un moyen simple, évolutif et flexible de gérer les autorisations. Toutefois, s'il n'est pas correctement conçu et géré, il peut autoriser ou refuser l'accès à vos clés KMS par inadvertance. Si vous utilisez des balises pour contrôler l'accès, tenez compte des pratiques suivantes.

- Utilisez des balises pour renforcer la bonne pratique de l'accès le moins privilégié. Accordez uniquement aux principaux IAM les autorisations dont ils ont besoin sur les clés KMS qu'ils doivent utiliser ou gérer. Par exemple, utilisez des balises pour étiqueter les clés KMS utilisées pour un projet. Donnez ensuite à l'équipe de projet l'autorisation d'utiliser uniquement les clés KMS avec la balise de projet.
- Soyez prudent lorsque vous donnez aux principaux les autorisations `kms:TagResource` et `kms:UntagResource` qui leur permettent d'ajouter, de modifier et de supprimer des balises. Lorsque vous utilisez des balises pour contrôler l'accès aux clés KMS, la modification d'une balise peut donner aux principaux l'autorisation d'utiliser des clés KMS qu'ils n'avaient alors pas l'autorisation d'utiliser. Elle peut également refuser l'accès aux clés KMS dont d'autres principaux ont besoin pour réaliser leurs tâches. Les administrateurs de clés qui n'ont pas l'autorisation de modifier les politiques de clé ou de créer des octrois peuvent contrôler l'accès aux clés KMS s'ils sont autorisés à gérer les balises.

  Dans la mesure du possible, utilisez une condition de politique, telle que `aws:RequestTag/tag-key` ou `aws:TagKeys` pour limiter les autorisations d'étiquetage d'un principal à des balises ou des modèles de balises spécifiques sur des clés KMS particulières.
- Passez en revue les principes Compte AWS qui disposent actuellement d'autorisations de balisage et de débalisage et ajustez-les si nécessaire. Par exemple, la politique de clé par défaut pour les administrateurs de clés de la console inclut les autorisations `kms:TagResource` et `kms:UntagResource` sur cette clé KMS. Les politiques IAM peuvent autoriser les autorisations d'étiquetage et de désétiquetage sur toutes les clés KMS. Par exemple, la politique AWSKeyManagementServicePowerUsergérée permet aux principaux de baliser, de débaliser et de répertorier les balises sur toutes les clés KMS.
- Avant de définir une politique qui dépend d'une balise, passez en revue les balises figurant sur les clés KMS de votre Compte AWS. Assurez-vous que votre politique s'applique uniquement aux balises que vous avez l'intention d'inclure. Utilisez CloudTrail les journaux et les CloudWatch

alarmes pour vous avertir des modifications de balises susceptibles d'affecter l'accès à vos clés KMS.

- Les conditions de politique de balise utilisent la correspondance de modèles ; elles ne sont pas liées à une instance particulière d'une balise. Une politique qui utilise des clés de condition basées sur des balises affecte toutes les balises nouvelles et existantes qui correspondent au modèle. Si vous supprimez et recréez une balise qui correspond à une condition de politique, la condition s'applique à la nouvelle balise, comme elle l'a fait pour l'ancienne.

Prenons l'exemple de la politique IAM suivante : Il permet aux principaux d'appeler les opérations GenerateDataKeyWithoutPlaintextet de déchiffrer uniquement sur les clés KMS de votre compte appartenant à la région Asie-Pacifique (Singapour) et dotées d'un tag. `"Project"="Alpha"` Vous pouvez attacher cette politique à des rôles dans l'exemple de projet Alpha.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:ap-southeast-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

L'exemple de politique IAM suivant autorise les principaux à utiliser n'importe quelle clé KMS dans le compte pour les opérations de chiffrement. Mais il interdit aux principaux d'utiliser ces opérations de chiffrement sur les clés KMS avec une identification `"Type"="Reserved"` ou sans identification `"Type"`.

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

```
  "Statement": [
    {
      "Sid": "IAMAllowCryptographicOperations",
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    },
    {
      "Sid": "IAMDenyOnTag",
      "Effect": "Deny",
      "Action": [
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Type": "Reserved"
        }
      }
    },
    {
      "Sid": "IAMDenyNoTag",
      "Effect": "Deny",
      "Action": [
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "Null": {
          "aws:ResourceTag/Type": "true"
        }
      }
    }
```

```
    ]
}
```

Principaux magasins

Un magasin de clés est un emplacement sécurisé pour le stockage et l'utilisation de clés cryptographiques. Le magasin de clés par défaut prend AWS KMS également en charge les méthodes de génération et de gestion des clés qu'il stocke. Par défaut, le contenu de la clé cryptographique dans AWS KMS keys lequel vous créez AWS KMS est généré et protégé par des modules de sécurité matériels (HSMs) qui sont le programme de validation des modules cryptographiques FIPS 140-3. Les éléments clés de vos clés KMS ne sont jamais HSMs non chiffrés.

AWS KMS prend en charge plusieurs types de magasins de clés pour protéger le contenu de vos clés lorsque vous AWS KMS les utilisez pour créer et gérer vos clés de chiffrement. Toutes les options de stockage de clés proposées par AWS KMS sont continuellement validées selon la norme FIPS 140-3 au niveau de sécurité 3 et sont conçues pour empêcher quiconque, y compris AWS les opérateurs, d'accéder à vos clés en texte brut ou de les utiliser sans votre autorisation.

AWS KMS magasin de clés standard

Par défaut, une clé KMS est créée à l'aide du AWS KMS HSM standard. Ce type de HSM peut être considéré comme un parc multi-locataires HSMs qui permet de disposer du magasin de clés le plus évolutif, le moins coûteux et le plus simple à gérer de votre point de vue. Si vous créez une clé KMS à utiliser dans un ou plusieurs services Services AWS afin que le service puisse chiffrer vos données en votre nom, vous allez créer une clé symétrique. Si vous utilisez une clé KMS pour la conception de votre propre application, vous pouvez choisir de créer une clé de chiffrement symétrique, une clé asymétrique ou une clé HMAC.

Dans l'option de stockage de clés standard, AWS KMS crée votre clé, puis la chiffre sous des clés gérées en interne par le service. Plusieurs copies des versions cryptées de vos clés sont ensuite stockées dans des systèmes conçus pour durer. La génération et la protection de votre matériel clé dans le type de magasin de clés standard vous permettent de tirer pleinement parti de l'évolutivité, de la disponibilité et de la durabilité des magasins AWS clés AWS KMS avec les charges opérationnelles et les coûts les plus faibles.

AWS KMS magasin de clés standard avec matériel de clé importé

Au lieu de demander AWS KMS à la fois de générer et de stocker les seules copies d'une clé donnée, vous pouvez choisir d'importer le contenu de la clé AWS KMS, ce qui vous permet

de générer votre propre clé de chiffrement symétrique de 256 bits, une clé RSA ou à courbe elliptique (ECC) ou une clé HMAC (Hash-Based Message Authentication Code), et de l'appliquer à un identifiant de clé KMS (KeyID). C'est ce que l'on appelle parfois « apportez votre propre clé » (BYOK). Les éléments clés importés depuis votre système de gestion de clés local doivent être protégés à l'aide d'une clé publique émise par AWS KMS, d'un algorithme d'encapsulage cryptographique pris en charge et d'un jeton d'importation basé sur le temps fourni par. AWS KMS Ce processus vérifie que votre clé cryptée et importée ne peut être déchiffrée par un AWS KMS HSM qu'une fois qu'elle a quitté votre environnement.

Le matériel clé importé peut être utile si vous avez des exigences spécifiques concernant le système qui génère les clés, ou si vous souhaitez une copie de votre clé à l'extérieur AWS comme sauvegarde. Notez que vous êtes responsable de la disponibilité et de la durabilité globales d'un matériau clé importé. Bien qu'il AWS KMS dispose d'une copie de votre clé importée et qu'il restera hautement disponible lorsque vous en aurez besoin, les clés importées offrent une API spéciale pour la suppression — DeleteImportedKeyMaterial. Cette API supprimera immédiatement toutes les copies du matériel clé importé qui en AWS KMS possède, sans possibilité AWS de récupérer la clé. En outre, vous pouvez définir une date d'expiration pour une clé importée, après laquelle la clé sera inutilisable. Pour que la clé soit à nouveau utile dans AWS KMS, vous devrez réimporter le contenu clé et l'attribuer au même KeyID. Cette action de suppression des clés importées est différente de celle des clés standard AWS KMS générées et stockées pour vous en votre nom. Dans le cas standard, le processus de suppression des clés comporte une période d'attente obligatoire au cours de laquelle l'utilisation d'une clé dont la suppression est prévue est d'abord bloquée. Cette action vous permet de voir les erreurs de refus d'accès dans les journaux de toute application ou AWS service susceptible d'avoir besoin de cette clé pour accéder aux données. Si vous recevez de telles demandes d'accès, vous pouvez choisir d'annuler la suppression planifiée et de réactiver la clé. Après une période d'attente configurable (entre 7 et 30 jours), ce n'est qu'alors que KMS supprimera réellement le contenu clé, le KeyID et toutes les métadonnées associées à la clé. Pour plus d'informations sur la disponibilité et la durabilité, consultez la section Protection du matériel clé importé dans le Guide du AWS KMS développeur.

Il convient de prendre en compte certaines limites supplémentaires liées au matériel clé importé. Comme il est AWS KMS impossible de générer de nouveaux éléments clés, il n'existe aucun moyen de configurer la rotation automatique des clés importées. Vous devrez créer une nouvelle clé KMS avec un nouveau KeyID, puis importer de nouveaux éléments clés pour obtenir une rotation efficace. De plus, les textes chiffrés créés AWS KMS sous une clé symétrique importée ne peuvent pas être facilement déchiffrés à l'aide de votre copie locale de la clé extérieure à. AWS Cela est dû au fait que le format de chiffrement authentifié utilisé par AWS KMS ajoute des métadonnées supplémentaires

au texte chiffré pour garantir lors de l'opération de déchiffrement que le texte chiffré a été créé par la clé KMS attendue lors d'une opération de chiffrement précédente. La plupart des systèmes cryptographiques externes ne comprendront pas comment analyser ces métadonnées pour accéder au texte chiffré brut afin de pouvoir utiliser leur copie d'une clé symétrique. Les textes chiffrés créés AWS KMS avec des clés asymétriques importées (par exemple RSA ou ECC) peuvent être utilisés en dehors de la partie correspondante (publique ou privée) de la clé car aucune métadonnée supplémentaire n'est ajoutée au texte chiffré. AWS KMS

AWS KMS magasins de clés personnalisés

Toutefois, si vous avez besoin d'un contrôle encore plus poussé HSMs, vous pouvez créer un magasin de clés personnalisé.

Un magasin de clés personnalisé est un magasin de clés AWS KMS intégré qui est soutenu par un gestionnaire de clés extérieur AWS KMS, que vous possédez et gérez. Les magasins de clés personnalisés combinent l'interface de gestion des clés pratique et complète AWS KMS avec la capacité de posséder et de contrôler le matériel clé et les opérations cryptographiques. Lorsque vous utilisez une clé KMS dans un magasin de clés personnalisé, les opérations cryptographiques sont effectuées par votre gestionnaire de clés en utilisant vos clés cryptographiques. Par conséquent, vous assumez une plus grande responsabilité quant à la disponibilité et à la durabilité des clés cryptographiques, ainsi qu'au fonctionnement des HSMs.

Posséder le vôtre HSMs peut être utile pour répondre à certaines exigences réglementaires qui n'autorisent pas encore les services Web mutualisés tels que le magasin de clés KMS standard à détenir vos clés cryptographiques. Les magasins de clés personnalisés ne sont pas plus sécurisés que les magasins de clés KMS gérés par AWS-managed HSMs, mais ils ont des implications différentes (et supérieures) en termes de gestion et de coûts. Par conséquent, vous assumez une plus grande responsabilité quant à la disponibilité et à la durabilité des clés cryptographiques ainsi qu'au fonctionnement des HSMs. Que vous utilisiez le magasin de clés standard AWS KMS HSMs ou un magasin de clés personnalisé, le service est conçu de telle sorte que personne, y compris les AWS employés, ne puisse récupérer vos clés en texte clair ou les utiliser sans votre autorisation. AWS KMS prend en charge deux types de magasins de clés personnalisés, les magasins de AWS CloudHSM clés et les magasins de clés externes.

Fonctions non prises en charge

AWS KMS ne prend pas en charge les fonctionnalités suivantes dans les magasins de clés personnalisés.

- Clés KMS asymétriques
- Clés KMS HMAC
- Clés KMS avec des éléments de clé importés
- Rotation automatique des clés
- Clés multi-région

AWS CloudHSM magasin de clés

Vous pouvez créer une clé KMS dans un magasin de AWS CloudHSMclés, où les clés utilisateur root sont générées, stockées et utilisées dans un AWS CloudHSM cluster que vous possédez et gérez. Les demandes AWS KMS d'utilisation d'une clé pour une opération cryptographique sont transmises à votre AWS CloudHSM cluster pour effectuer l'opération. Bien qu'un AWS CloudHSM cluster soit hébergé par AWS, il s'agit d'une solution à locataire unique que vous gérez et exploitez directement. Vous êtes le principal responsable de la disponibilité et des performances des clés KMS dans un AWS CloudHSM cluster. Pour savoir si un magasin de clés AWS CloudHSM personnalisé répond à vos besoins, lisez l'article Les magasins de clés AWS KMS personnalisés vous conviennent-ils ? sur le blog consacré à AWS la sécurité.

Magasin de clés externe

Vous pouvez configurer AWS KMS pour utiliser un magasin de clés externe (XKS), dans lequel les clés de l'utilisateur root sont générées, stockées et utilisées dans un système de gestion des clés extérieur au AWS Cloud. Les demandes AWS KMS d'utilisation d'une clé pour une opération cryptographique sont transmises à votre système hébergé en externe pour effectuer l'opération. Plus précisément, les demandes sont transmises à un proxy XKS de votre réseau, qui les transmet ensuite au système cryptographique que vous utilisez. Le proxy XKS est une spécification open source à laquelle tout le monde peut s'intégrer. De nombreux fournisseurs commerciaux de gestion de clés prennent en charge la spécification XKS Proxy. Comme un magasin de clés externe est hébergé par vous ou par un tiers, vous êtes responsable de la disponibilité, de la durabilité et des performances des clés du système. Pour savoir si un magasin de clés externe répond à vos besoins, lisez Announding AWS KMS External Key Store (XKS) sur le blog AWS News.

AWS CloudHSM magasins clés

Un magasin de AWS CloudHSM clés est un magasin de clés personnalisé soutenu par un AWS CloudHSM cluster. Lorsque vous créez un magasin AWS KMS key de clés personnalisé, vous

AWS KMS générez et stockez des éléments clés non extractibles pour la clé KMS dans un AWS CloudHSM cluster que vous possédez et gérez. Lorsque vous utilisez une clé KMS dans un magasin de clés personnalisé, les opérations cryptographiques sont effectuées HSMs dans le cluster. Cette fonctionnalité combine la commodité et AWS KMS l'intégration généralisée d'un AWS CloudHSM cluster dans votre Compte AWS.

AWS KMS fournit un support complet de console et d'API pour la création, l'utilisation et la gestion de vos magasins de clés personnalisés. Vous pouvez utiliser les clés KMS dans votre magasin de clés personnalisé de la même manière que vous utilisez n'importe quelle clé KMS. Par exemple, vous pouvez utiliser les clés KMS pour générer des clés de données et chiffrer des données. Vous pouvez également utiliser les clés KMS dans votre magasin de clés personnalisé avec des AWS services prenant en charge les clés gérées par le client.

Est-ce que j'ai besoin d'un magasin de clés personnalisé ?

Pour la plupart des utilisateurs, le magasin de AWS KMS clés par défaut, qui est protégé par des modules cryptographiques validés par la norme FIPS 140-3, répond à leurs exigences de sécurité. Il n'est pas nécessaire d'ajouter une couche supplémentaire de responsabilité de maintenance ou une dépendance à l'égard d'un service supplémentaire.

Cependant, vous pouvez envisager la création d'un magasin de clés personnalisé si votre organisation possède l'une des exigences suivantes :

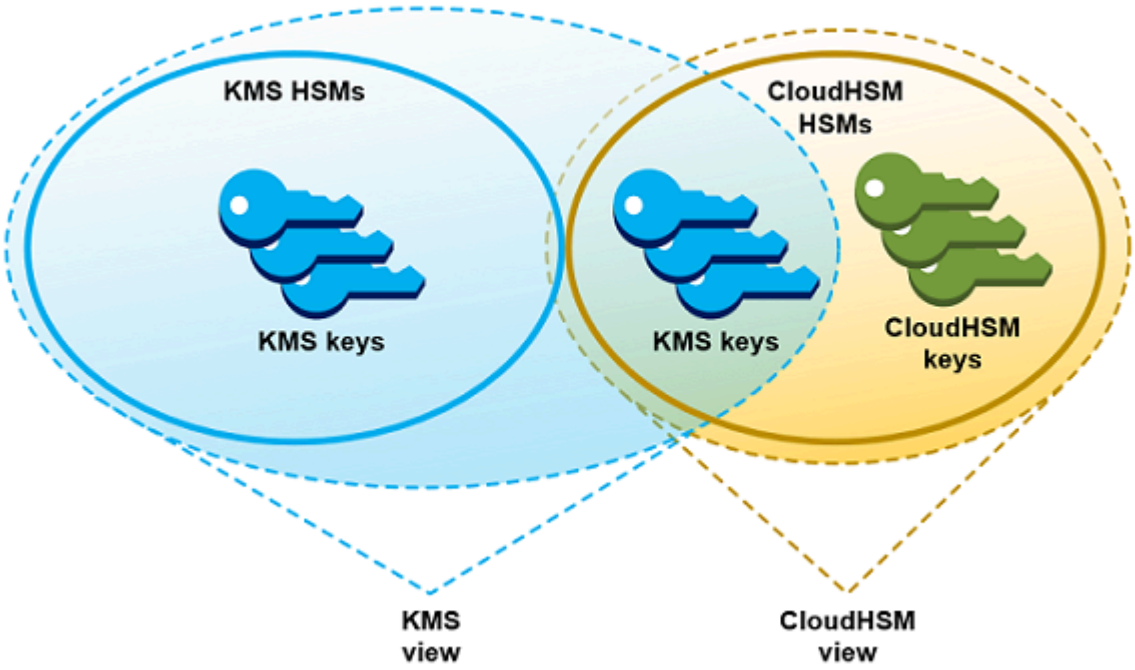
- Vous avez des clés qui doivent explicitement être protégées dans un HSM à locataire unique ou dans un HSM sur lequel vous avez un contrôle direct.
- Vous devez être en mesure de retirer immédiatement les éléments clés de AWS KMS.
- Vous devez être en mesure d'auditer toute utilisation de vos clés indépendamment de AWS KMS ou AWS CloudTrail.

Comment fonctionnent les magasins de clés personnalisés ?

Chaque magasin de clés personnalisé est associé à un AWS CloudHSM cluster dans votre Compte AWS. Lorsque vous connectez le magasin de clés personnalisé à son cluster, il AWS KMS crée l'infrastructure réseau pour prendre en charge la connexion. Il se connecte ensuite au AWS CloudHSM client clé du cluster à l'aide des informations d'identification d'un utilisateur cryptographique dédié du cluster.

Vous créez et gérez vos magasins de clés personnalisés dans AWS KMS et vous créez et gérez vos clusters HSM dans AWS CloudHSM. Lorsque vous créez AWS KMS keys dans un magasin de clés

AWS KMS personnalisé, vous visualisez et gérez les clés KMS dans AWS KMS. Mais vous pouvez également afficher et gérer leurs clés dans AWS CloudHSM, tout comme vous le feriez pour d'autres clés du cluster.



Vous pouvez créer des clés KMS de chiffrement symétriques à partir du contenu clé généré par AWS KMS votre magasin de clés personnalisé. Utilisez ensuite les mêmes techniques pour afficher et gérer les clés KMS dans votre magasin de clés personnalisé que celles que vous utilisez pour les clés KMS dans le magasin de AWS KMS clés. Vous pouvez contrôler l'accès aux politiques IAM et aux politiques de clé, créer des balises et des alias, activer et désactiver les clés KMS, et planifier la suppression de clés. Vous pouvez utiliser les clés KMS pour des opérations cryptographiques et les utiliser avec AWS des services intégrés à AWS KMS.

En outre, vous avez le contrôle total du AWS CloudHSM cluster, y compris la création, la suppression HSMs et la gestion des sauvegardes. Vous pouvez utiliser le AWS CloudHSM client et les bibliothèques logicielles prises en charge pour visualiser, auditer et gérer les éléments clés de vos clés KMS. Lorsque le magasin de clés personnalisé est déconnecté, il AWS KMS ne peut pas y accéder et les utilisateurs ne peuvent pas utiliser les clés KMS du magasin de clés personnalisé pour des opérations cryptographiques. Cette nouvelle couche de contrôle fait du magasin de clés personnalisé une solution puissante pour les organisations qui en ont besoin.

Où commencer ?

Pour créer et gérer un magasin de AWS CloudHSM clés, vous utilisez les fonctionnalités de AWS KMS et AWS CloudHSM.

1. Commencez par AWS CloudHSMCréez un cluster AWS CloudHSM actif ou sélectionnez un cluster existant. Le cluster doit en avoir au moins deux actifs HSMs dans des zones de disponibilité différentes. Ensuite, créez un compte CU (utilisateur de chiffrement) dédié dans ce cluster pour AWS KMS.
2. Dans AWS KMS, créez un magasin de clés personnalisé associé au AWS CloudHSM cluster sélectionné. AWS KMS fournit une interface de gestion complète qui vous permet de créer, d'afficher, de modifier et de supprimer vos banques de clés personnalisées.
3. Lorsque vous êtes prêt à utiliser votre magasin de clés personnalisé, connectez-le au AWS CloudHSM cluster associé. AWS KMS crée l'infrastructure réseau dont elle a besoin pour prendre en charge la connexion. Ensuite, il se connecte au cluster à l'aide des informations d'identification du compte CU dédié afin de générer et de gérer les clés dans le cluster.
4. Vous pouvez désormais créer des clés KMS de chiffrement symétriques dans votre magasin de clés personnalisé. Il vous suffit de spécifier le magasin de clés personnalisé lorsque vous créez la clé KMS.

Si vous vous retrouvez bloqué à un moment, vous pouvez obtenir de l'aide dans la rubrique Dépannage d'un magasin de clés personnalisé. Si vous ne trouvez pas de réponse à votre question, utilisez le lien situé en bas de chaque page de ce guide ou publiez une question sur le AWS Key Management Service forum de discussion.

Quotas

AWS KMS autorise jusqu'à 10 magasins de clés personnalisés dans chaque Compte AWS région, y compris les magasins de AWS CloudHSM clés et les magasins de clés externes, quel que soit leur état de connexion. En outre, il existe des quotas de AWS KMS demandes concernant l'utilisation des clés KMS dans un magasin de AWS CloudHSM clés.

Tarification

Pour plus d'informations sur le coût des magasins de clés AWS KMS personnalisés et des clés gérées par le client dans un magasin de clés personnalisé, consultez AWS Key Management Service les tarifs. Pour plus d'informations sur le coût des AWS CloudHSM clusters HSMs, consultez la section AWS CloudHSM Tarification.

Régions

AWS KMS prend en charge les AWS CloudHSM principaux magasins dans Régions AWS tous AWS KMS les pays concernés, à l'exception de l'Asie-Pacifique (Melbourne), de la Chine (Pékin), de la Chine (Ningxia) et de l'Europe (Espagne).

Fonctions non prises en charge

AWS KMS ne prend pas en charge les fonctionnalités suivantes dans les magasins de clés personnalisés.

- Clés KMS asymétriques
- Clés KMS HMAC
- Clés KMS avec des éléments de clé importés
- Rotation automatique des clés
- Clés multi-région

AWS CloudHSM concepts clés du magasin

Cette rubrique explique certains termes et concepts utilisés dans les AWS CloudHSM principaux magasins.

AWS CloudHSM magasin de clés

Un magasin de AWS CloudHSM clés est un magasin de clés personnalisé associé à un AWS CloudHSM cluster que vous possédez et gérez. AWS CloudHSM les clusters sont soutenus par des modules de sécurité matériels (HSMs) certifiés FIPS 140-2 ou FIPS 140-3 de niveau 3.

Lorsque vous créez une clé KMS dans votre magasin de AWS CloudHSM clés, elle AWS KMS génère une clé symétrique AES (Advanced Encryption Standard) de 256 bits, persistante et non exportable dans le cluster associé. AWS CloudHSM Ce matériel clé ne vous laisse jamais HSMs déchiffré. Lorsque vous utilisez une clé KMS dans un magasin de AWS CloudHSM clés, les opérations cryptographiques sont effectuées HSMs dans le cluster.

AWS CloudHSM les magasins de clés associent l'interface de gestion des clés pratique et complète AWS KMS aux commandes supplémentaires fournies par un AWS CloudHSM cluster intégré à votre Compte AWS. Cette fonctionnalité intégrée vous permet de créer, de gérer et d'utiliser des clés KMS AWS KMS tout en gardant le contrôle total sur ceux HSMs qui stockent leurs informations clés, y compris la gestion des clusters et des sauvegardes. HSMs Vous pouvez utiliser la AWS KMS console

et APIs gérer le magasin de AWS CloudHSM clés et ses clés KMS. Vous pouvez également utiliser la AWS CloudHSM console APIs, le logiciel client et les bibliothèques de logiciels associées pour gérer le cluster associé.

Vous pouvez afficher et gérer votre magasin de AWS CloudHSM clés, modifier ses propriétés, le connecter et le déconnecter de son AWS CloudHSM cluster associé. Si vous devez supprimer un magasin de AWS CloudHSM clés, vous devez d'abord supprimer les clés KMS du AWS CloudHSM magasin de clés en programmant leur suppression et en attendant l'expiration du délai de grâce. La suppression du magasin de AWS CloudHSM clés entraîne la suppression de la ressource AWS KMS, mais cela n'affecte pas votre AWS CloudHSM cluster.

AWS CloudHSM grappe

Chaque magasin de AWS CloudHSM clés est associé à un AWS CloudHSM cluster. Lorsque vous créez un élément AWS KMS key dans votre magasin de AWS CloudHSM clés, il AWS KMS crée son contenu clé dans le cluster associé. Lorsque vous utilisez une clé KMS dans votre magasin de clés AWS CloudHSM , les opérations cryptographiques sont effectuées dans le cluster associé.

Chaque AWS CloudHSM cluster ne peut être associé qu'à un seul magasin de AWS CloudHSM clés. Le cluster que vous choisissez ne peut pas être associé à un autre magasin de AWS CloudHSM clés ni partager un historique de sauvegarde avec un cluster associé à un autre magasin de AWS CloudHSM clés. Le cluster doit être initialisé et actif, et il doit se trouver dans la même Compte AWS région que le magasin de AWS CloudHSM clés. Vous pouvez créer un nouveau cluster ou utiliser un cluster existant. AWS KMS ne nécessite pas l'utilisation exclusive du cluster. Pour créer des clés KMS dans le magasin de AWS CloudHSM clés, son cluster associé doit contenir au moins deux clés actives HSMs. Toutes les autres opérations nécessitent un seul HSM.

Vous spécifiez le AWS CloudHSM cluster lorsque vous créez le magasin de AWS CloudHSM clés, et vous ne pouvez pas le modifier. Cependant, vous pouvez remplacer n'importe quel cluster qui partage un historique de sauvegardes avec le cluster d'origine. Cela vous permet de supprimer le cluster, si nécessaire, et de le remplacer-le par un cluster créé à partir de l'une de ses sauvegardes. Vous conservez le contrôle total du AWS CloudHSM cluster associé, ce qui vous permet de gérer les utilisateurs et les clés, de créer et de supprimer HSMs, ainsi que d'utiliser et de gérer des sauvegardes.

Lorsque vous êtes prêt à utiliser votre magasin de AWS CloudHSM clés, vous le connectez au AWS CloudHSM cluster associé. Vous pouvez connecter et déconnecter votre magasin de clés personnalisé à tout moment. Lorsqu'un magasin de clés personnalisé est connecté, vous pouvez créer et utiliser ses clés KMS. Lorsqu'il est déconnecté, vous pouvez consulter et gérer le magasin de

AWS CloudHSM clés et ses clés KMS. Toutefois, vous ne pouvez pas créer de nouvelles clés KMS ni utiliser les clés KMS du magasin de AWS CloudHSM clés pour des opérations cryptographiques.

Utilisateur de chiffrement **`kmsuser`**

Pour créer et gérer des éléments clés dans le AWS CloudHSM cluster associé en votre nom, AWS KMS utilisez un utilisateur AWS CloudHSM cryptographique (CU) dédié dans le cluster nommékmsuser. Le kmsuser CU est un compte CU standard qui est automatiquement synchronisé avec tous les membres HSMs du cluster et enregistré dans les sauvegardes du cluster.

Avant de créer votre magasin de AWS CloudHSM clés, vous devez créer un compte kmsuser CU dans votre AWS CloudHSM cluster à l'aide de la commande user create de la CLI CloudHSM. Ensuite, lorsque vous créez le magasin de AWS CloudHSM clés, vous fournissez le mot de passe du kmsuser compte à AWS KMS. Lorsque vous connectez le magasin de clés personnalisé, vous AWS KMS vous connectez au cluster en tant que kmsuser CU et changez son mot de passe. AWS KMS chiffre votre kmsuser mot de passe avant de le stocker en toute sécurité. Lorsque le mot de passe a effectué une rotation, le nouveau mot de passe est chiffré et stocké de la même manière.

AWS KMS reste connecté kmsuser tant que le magasin de AWS CloudHSM clés est connecté. Vous ne devez pas utiliser ce compte CU à d'autres fins. Toutefois, vous gardez le contrôle ultime du compte CU kmsuser. À tout moment, vous pouvez retrouver les clés qui en sont kmsuser propriétaires. Si nécessaire, vous pouvez déconnecter le magasin de clés personnalisé, modifier le mot de passe kmsuser, vous connecter au cluster en tant que kmsuser et afficher et gérer les clés que kmsuser détient.

Pour obtenir des instructions sur la création de votre compte CU kmsuser, consultez Créer l'utilisateur de chiffrement (CU) kmsuser.

Clés KMS dans un magasin de AWS CloudHSM clés

Vous pouvez utiliser l' AWS KMS API AWS KMS or pour créer un AWS KMS keys dans un magasin de AWS CloudHSM clés. Vous utilisez la même technique que celle que vous utiliseriez sur n'importe quelle clé KMS. La seule différence est que vous devez identifier le magasin de AWS CloudHSM clés et spécifier que l'origine du matériau clé est le AWS CloudHSM cluster.

Lorsque vous créez une clé KMS dans un magasin de AWS CloudHSM clés, vous AWS KMS créez la clé KMS dans AWS KMS et celle-ci génère une clé symétrique AES (Advanced Encryption Standard) de 256 bits, persistante et non exportable dans le cluster associé. Lorsque vous utilisez la AWS KMS clé dans une opération cryptographique, l'opération est effectuée dans le cluster à l'aide

de la clé AES basée sur le AWS CloudHSM cluster. Bien qu' AWS CloudHSM ils prennent en charge les clés symétriques et asymétriques de différents types, les magasins de clés ne prennent en charge que les AWS CloudHSM clés de chiffrement symétriques AES.

Vous pouvez afficher les clés KMS dans un magasin de AWS CloudHSM clés de la AWS KMS console et utiliser les options de la console pour afficher l'ID de magasin de clés personnalisé. Vous pouvez également utiliser cette DescribeKeyopération pour trouver l'ID du magasin de AWS CloudHSM clés et l'ID AWS CloudHSM du cluster.

Les clés KMS d'un magasin de AWS CloudHSM clés fonctionnent comme n'importe quelle clé KMS dans un magasin de clés AWS KMS. Les utilisateurs autorisés ont besoin des mêmes autorisations pour utiliser et gérer les clés KMS. Vous utilisez les mêmes procédures de console et les mêmes opérations d'API pour afficher et gérer les clés KMS dans un magasin de AWS CloudHSM clés. Cela inclut l'activation et la désactivation de clés KMS, la création et l'utilisation de balises et d'alias, et la définition et la modification des politiques de clé et des politiques IAM. Vous pouvez utiliser les clés KMS d'un magasin de AWS CloudHSM clés pour des opérations cryptographiques et les utiliser avec des AWS services intégrés qui prennent en charge l'utilisation de clés gérées par le client. Cependant, vous ne pouvez pas activer la rotation automatique des clés ni importer des éléments clés dans une clé KMS dans un magasin de AWS CloudHSM clés.

Vous utilisez également le même processus pour planifier la suppression d'une clé KMS dans un magasin de AWS CloudHSM clés. Une fois le délai d'attente expiré, AWS KMS supprime la clé KMS de KMS. Il fait ensuite de son mieux pour supprimer le contenu clé de la clé KMS du AWS CloudHSM cluster associé. Cependant, il se peut que vous ayez besoin de supprimer manuellement les éléments de clé orphelins du cluster et de ses sauvegardes.

Contrôlez l'accès à votre magasin de AWS CloudHSM clés

Vous utilisez des politiques IAM pour contrôler l'accès à votre magasin de AWS CloudHSM clés et à votre AWS CloudHSM cluster. Vous pouvez utiliser des politiques clés, des politiques IAM et des autorisations pour contrôler l'accès AWS KMS keys à votre magasin de AWS CloudHSM clés. Nous vous recommandons de fournir aux utilisateurs, groupes et rôles uniquement les autorisations dont ils ont besoin pour les tâches qu'ils sont susceptibles d'effectuer.

Pour prendre en charge vos AWS CloudHSM principaux magasins, vous AWS KMS avez besoin d'une autorisation pour obtenir des informations sur vos AWS CloudHSM clusters. Il a également besoin d'une autorisation pour créer l'infrastructure réseau qui connecte votre magasin de AWS CloudHSM clés à son AWS CloudHSM cluster. Pour obtenir ces autorisations, AWS KMS crée

le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié au service dans votre. Compte AWS Pour de plus amples informations, veuillez consulter Autorisation AWS KMS de gestion AWS CloudHSM et ressources Amazon EC2 .

Lorsque vous concevez votre magasin de AWS CloudHSM clés, assurez-vous que les principaux responsables qui l'utilisent et le gèrent disposent uniquement des autorisations dont ils ont besoin. La liste suivante décrit les autorisations minimales requises pour les AWS CloudHSM principaux responsables de magasins et utilisateurs.

- Les responsables qui créent et gèrent votre magasin de AWS CloudHSM clés ont besoin de l'autorisation suivante pour utiliser les opérations de l'API du magasin de AWS CloudHSM clés.
  - `cloudhsm:DescribeClusters`
  - `kms:CreateCustomKeyStore`
  - `kms:ConnectCustomKeyStore`
  - `kms:DeleteCustomKeyStore`
  - `kms:DescribeCustomKeyStores`
  - `kms:DisconnectCustomKeyStore`
  - `kms:UpdateCustomKeyStore`
  - `iam:CreateServiceLinkedRole`
- Les principaux responsables qui créent et gèrent le AWS CloudHSM cluster associé à votre banque de AWS CloudHSM clés doivent être autorisés à créer et à initialiser un AWS CloudHSM cluster. Cela inclut l'autorisation de créer ou d'utiliser un Amazon Virtual Private Cloud (VPC), de créer des sous-réseaux et de créer une instance Amazon. EC2 Ils peuvent également avoir besoin de créer HSMs, de supprimer et de gérer des sauvegardes. Pour obtenir la liste des autorisations requises, veuillez consulter la rubrique Identity and access management for AWS CloudHSM (Gestion des identités et des accès pour) dans le Guide de l'utilisateur AWS CloudHSM .
- Les principaux responsables qui créent et gèrent AWS KMS keys dans votre magasin de AWS CloudHSM clés ont besoin des mêmes autorisations que ceux qui créent et gèrent n'importe quelle clé KMS dans AWS KMS votre magasin de clés. La politique de clé par défaut pour une clé KMS dans un magasin de AWS CloudHSM clés est identique à la politique de clé par défaut pour les clés KMS dans AWS KMS. Le contrôle d'accès basé sur les attributs (ABAC), qui utilise des balises et des alias pour contrôler l'accès aux clés KMS, est également efficace sur les clés KMS dans les magasins de clés. AWS CloudHSM
- Les principaux qui utilisent les clés KMS de votre AWS CloudHSM magasin de clés pour des opérations cryptographiques doivent être autorisés à effectuer l'opération cryptographique avec la

clé KMS, telle que KMS:Decrypt. Vous pouvez fournir ces autorisations dans une politique de clé, ou une politique IAM. Toutefois, ils n'ont pas besoin d'autorisations supplémentaires pour utiliser une clé KMS dans un magasin de AWS CloudHSM clés.

Création d'un magasin de AWS CloudHSM clés

Vous pouvez créer un ou plusieurs magasins AWS CloudHSM clés dans votre compte. Chaque magasin de AWS CloudHSM clés est associé à un AWS CloudHSM cluster de la même Compte AWS région. Avant de créer votre magasin de clés AWS CloudHSM , vous devez réunir les conditions préalables. Ensuite, avant de pouvoir utiliser votre magasin de AWS CloudHSM clés, vous devez le connecter à son AWS CloudHSM cluster.

> **Remarques**
>
> KMS ne peut pas communiquer IPv6 avec les magasins de AWS CloudHSM clés.
> Si vous essayez de créer un magasin de AWS CloudHSM clés avec toutes les mêmes valeurs de propriétés qu'un magasin de AWS CloudHSM clés déconnecté existant, AWS KMS cela ne crée pas de nouveau magasin de AWS CloudHSM clés et ne génère aucune exception ni n'affiche d'erreur. AWS KMS Reconnaît plutôt le doublon comme la conséquence probable d'une nouvelle tentative et renvoie l'ID du magasin de AWS CloudHSM clés existant.
> Il n'est pas nécessaire de connecter immédiatement votre magasin de AWS CloudHSM clés. Vous pouvez le conserver dans un état déconnecté jusqu'à ce que vous soyez prêt à l'utiliser. Cependant, afin de vérifier qu'il est correctement configuré, vous pouvez le connecter, afficher son état de connexion, puis le déconnecter.

Rubriques

- Rassembler les conditions requises
- Création d'un nouveau magasin de AWS CloudHSM clés

Rassembler les conditions requises

Chaque magasin de AWS CloudHSM clés est soutenu par un AWS CloudHSM cluster. Pour créer un magasin de AWS CloudHSM clés, vous devez spécifier un AWS CloudHSM cluster actif qui n'est pas déjà associé à un autre magasin de clés. Vous devez également créer un utilisateur cryptographique

(CU) dédié dans le cluster HSMs qui AWS KMS peut l'utiliser pour créer et gérer des clés en votre nom.

Avant de créer un magasin de AWS CloudHSM clés, procédez comme suit :

Sélectionnez un AWS CloudHSM cluster

Chaque magasin de AWS CloudHSM clés est associé à un seul AWS CloudHSM cluster. Lorsque vous créez AWS KMS keys dans votre magasin de AWS CloudHSM clés, vous AWS KMS créez les métadonnées de la clé KMS, telles qu'un identifiant et un Amazon Resource Name (ARN) dans AWS KMS. Il crée ensuite le matériau clé dans HSMs le cluster associé. Vous pouvez créer un nouveau AWS CloudHSM cluster ou utiliser un cluster existant. AWS KMS ne nécessite pas d'accès exclusif au cluster.

Le AWS CloudHSM cluster que vous sélectionnez est associé de façon permanente au magasin de AWS CloudHSM clés. Après avoir créé le magasin de AWS CloudHSM clés, vous pouvez modifier l'ID du cluster associé, mais le cluster que vous spécifiez doit partager un historique de sauvegarde avec le cluster d'origine. Pour utiliser un cluster indépendant, vous devez créer un nouveau magasin de AWS CloudHSM clés.

Le AWS CloudHSM cluster que vous sélectionnez doit présenter les caractéristiques suivantes :

- Le cluster doit être actif.

  Vous devez créer le cluster, l'initialiser, installer le logiciel AWS CloudHSM client pour votre plate-forme, puis activer le cluster. Pour plus d'informations, veuillez consulter la rubrique Getting started with AWS CloudHSM (Démarrer avec) dans le Guide de l'utilisateur AWS CloudHSM .
- Le cluster doit se trouver dans le même compte et dans la même région que le magasin de AWS CloudHSM clés. Vous ne pouvez pas associer un magasin de AWS CloudHSM clés d'une région à un cluster d'une autre région. Pour créer une infrastructure clé dans plusieurs régions, vous devez créer des magasins de AWS CloudHSM clés et des clusters dans chaque région.
- Le cluster ne peut pas être associé à un autre magasin de clés personnalisé à partir du même compte et de la même région. Chaque magasin de AWS CloudHSM clés du compte et de la région doit être associé à un AWS CloudHSM cluster différent. Vous ne pouvez pas spécifier un cluster qui est déjà associé à un magasin de clés personnalisé ou un cluster qui partage un historique des sauvegardes avec un cluster associé. Les clusters qui partagent un historique des sauvegardes historique ont le même certificat de cluster. Pour afficher le certificat de cluster d'un cluster, utilisez la AWS CloudHSM console ou l'DescribeClustersopération.

Si vous sauvegardez un cluster AWS CloudHSM dans une autre région, il est considéré comme un cluster différent et vous pouvez associer la sauvegarde à un magasin de clés personnalisé dans sa région. Cependant, les clés KMS des deux magasins de clés personnalisés ne sont pas interopérables, même si elles possèdent la même clé de sauvegarde. AWS KMS lie les métadonnées au texte chiffré afin qu'il ne puisse être déchiffré que par la clé KMS qui l'a chiffré.

- Le cluster doit être configuré avec des sous-réseaux privés dans au moins deux zones de disponibilité de la région. Comme il n' AWS CloudHSM est pas pris en charge dans toutes les zones de disponibilité, nous vous recommandons de créer des sous-réseaux privés dans toutes les zones de disponibilité de la région. Vous ne pouvez pas reconfigurer les sous-réseaux d'un cluster existant, mais vous pouvez créer un cluster à partir d'une sauvegarde avec différents sous-réseaux dans la configuration du cluster.

  > **Important**
  >
  > Après avoir créé votre magasin de AWS CloudHSM clés, ne supprimez aucun des sous-réseaux privés configurés pour son AWS CloudHSM cluster. Si vous AWS KMS ne trouvez pas tous les sous-réseaux dans la configuration du cluster, les tentatives de connexion au magasin de clés personnalisé échouent avec un état d'erreur de `SUBNET_NOT_FOUND` connexion. Pour en savoir plus, consultez Comment corriger un échec de connexion.

- Le groupe de sécurité du cluster (`cloudhsm-cluster-<cluster-id>-sg`) doit inclure des règles entrantes et sortantes qui autorisent le trafic TCP sur les ports IPv4 2223-2225. La source des règles entrantes et la destination des règles sortantes doit correspondre à l'ID du groupe de sécurité. Ces règles sont définies par défaut lorsque vous créez le cluster. Ne pas les supprimer ou les modifier.
- Le cluster doit contenir au moins deux actifs HSMs dans des zones de disponibilité différentes. Pour vérifier le nombre de HSMs, utilisez la AWS CloudHSM console ou l'DescribeClustersopération. Si nécessaire, vous pouvez ajouter un module HSM.

Recherche le certificat approuvé (ou « trust anchor »)

Lorsque vous créez un magasin de clés personnalisé, vous devez télécharger le certificat d'ancrage de confiance du AWS CloudHSM cluster sur AWS KMS. AWS KMS a besoin du certificat Trust Anchor pour connecter le magasin de AWS CloudHSM clés à son AWS CloudHSM cluster associé.

Chaque AWS CloudHSM cluster actif possède un certificat d'ancrage de confiance. Lorsque vous initialisez le cluster, vous devez générer ce certificat, l'enregistrer dans le fichier `customerCA.crt` et le copier sur les hôtes qui se connectent au cluster.

Créez l'utilisateur **`kmsuser`** cryptographique pour AWS KMS

Pour administrer votre magasin de AWS CloudHSM clés AWS KMS , connectez-vous au compte utilisateur `kmsuser` cryptographique (CU) du cluster sélectionné. Avant de créer votre magasin de AWS CloudHSM clés, vous devez créer le `kmsuser` CU. Ensuite, lorsque vous créez votre magasin de AWS CloudHSM clés, vous fournissez le mot de passe `kmsuser` pour AWS KMS. Chaque fois que vous connectez le magasin de AWS CloudHSM clés à son AWS CloudHSM cluster associé, connectez-vous AWS KMS en tant que `kmsuser` et alternez le mot de `kmsuser` passe

> **Important**
>
> Ne spécifiez pas l'option `2FA` lorsque vous créez l'`kmsuser`utilisateur de chiffrement. Si vous le faites, vous AWS KMS ne pouvez pas vous connecter et votre magasin de AWS CloudHSM clés ne peut pas être connecté à ce AWS CloudHSM cluster. Une fois que vous spécifiez 2FA, vous ne pouvez pas l'annuler. Vous devez à la place supprimer l'utilisateur de chiffrement et le recréer.

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

1. Suivez les procédures de démarrage décrites dans la rubrique Getting Started with CloudHSM Command Line Interface (CLI) du Guide de l'AWS CloudHSM utilisateur.
2. Utilisez la commande user create pour créer une CU nomméekmsuser.

Le mot de passe doit contenir entre 7 et 32 caractères alphanumériques. Il est sensible à la casse et ne peut contenir aucun des caractères spéciaux.

L'exemple de commande suivant crée une `kmsuser` CU.

```
aws-cloudhsm > user create --username kmsuser --role crypto-user
Enter password:
Confirm password:
{
 "error_code": 0,
 "data": {
   "username": "kmsuser",
   "role": "crypto-user"
 }
}
```

Création d'un nouveau magasin de AWS CloudHSM clés

Après avoir assemblé les prérequis, vous pouvez créer un nouveau magasin de AWS CloudHSM clés dans la AWS KMS console ou en utilisant l'CreateCustomKeyStoreopération.

Utilisation de la AWS KMS console

Lorsque vous créez un magasin de AWS CloudHSM clés dans le AWS Management Console, vous pouvez ajouter et créer les prérequis dans le cadre de votre flux de travail. Toutefois, le processus est plus rapide que vous les avez assemblées au préalable.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), AWS CloudHSM key stores (Magasins de clés).
4. Choisissez Créer un magasin de clés.
5. Entrez un nom convivial pour le magasin de clés personnalisé. Le nom doit être unique parmi tous les magasins de clés personnalisés de votre compte.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

6. Sélectionnez un AWS CloudHSM cluster pour le magasin de AWS CloudHSM clés. Ou, pour créer un nouveau AWS CloudHSM cluster, cliquez sur le lien Créer un AWS CloudHSM cluster.

   Le menu affiche les AWS CloudHSM clusters de votre compte et de votre région qui ne sont pas encore associés à un magasin de AWS CloudHSM clés. Le cluster doit respecter les exigences d'association à un magasin de clés personnalisé.
7. Choisissez Choisir un fichier, puis téléchargez le certificat d'ancrage de confiance pour le AWS CloudHSM cluster que vous avez choisi. Il s'agit du fichier `customerCA.crt` que vous avez créé lorsque vous avez initialisé le cluster.
8. Entrez le mot de passe de l'utilisateur de chiffrement `kmsuser` (CU) que vous avez créé dans le cluster sélectionné.
9. Choisissez Créer.

Lorsque la procédure aboutit, le nouveau magasin de AWS CloudHSM clés apparaît dans la liste des magasins de AWS CloudHSM clés du compte et de la région. S'il ne réussit pas, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez Dépannage d'un magasin de clés personnalisé.

Si vous essayez de créer un magasin de AWS CloudHSM clés avec toutes les mêmes valeurs de propriétés qu'un magasin de AWS CloudHSM clés déconnecté existant, AWS KMS cela ne crée pas de nouveau magasin de AWS CloudHSM clés et ne génère aucune exception ni n'affiche d'erreur. AWS KMS Reconnaît plutôt le doublon comme la conséquence probable d'une nouvelle tentative et renvoie l'ID du magasin de AWS CloudHSM clés existant.

Suivant : Les nouveaux magasins de AWS CloudHSM clés ne sont pas automatiquement connectés. Avant de pouvoir créer AWS KMS keys dans le magasin de AWS CloudHSM clés, vous devez connecter le magasin de clés personnalisé au AWS CloudHSM cluster associé.

Utilisation de l' AWS KMS API

Vous pouvez utiliser cette CreateCustomKeyStoreopération pour créer un nouveau magasin de AWS CloudHSM clés associé à un AWS CloudHSM cluster dans le compte et la région. Ces exemples

utilisent l' AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

L'opération `CreateCustomKeyStore` nécessite les valeurs de paramètre suivantes.

- CustomKeyStoreName — Un nom convivial pour le magasin de clés personnalisé, unique dans le compte.

  > **Important**
  >
  > N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

- CloudHsmClusterId — L'ID de cluster d'un AWS CloudHSM cluster répondant aux exigences d'un magasin de AWS CloudHSM clés.
- KeyStorePassword — Le mot de passe du compte `kmsuser` CU dans le cluster spécifié.
- TrustAnchorCertificate — Le contenu du `customerCA.crt` fichier que vous avez créé lors de l'initialisation du cluster.

L'exemple suivant utilise un ID de cluster fictif. Avant d'exécuter la commande, remplacez-le par un ID de cluster valide.

```
$ aws kms create-custom-key-store
        --custom-key-store-name ExampleCloudHSMKeyStore \
        --cloud-hsm-cluster-id cluster-1a23b4cdefg \
        --key-store-password kmsPswd \
        --trust-anchor-certificate <certificate-goes-here>
```

Si vous utilisez le AWS CLI, vous pouvez spécifier le fichier de certificat d'ancrage de confiance au lieu de son contenu. Dans l'exemple suivant, le fichier `customerCA.crt` se trouve dans le répertoire racine.

```
$ aws kms create-custom-key-store
        --custom-key-store-name ExampleCloudHSMKeyStore \
        --cloud-hsm-cluster-id cluster-1a23b4cdefg \
        --key-store-password kmsPswd \
        --trust-anchor-certificate file://customerCA.crt
```

Lorsque l'opération est réussie, `CreateCustomKeyStore` renvoie l'ID du magasin de clés personnalisé, comme illustré dans l'exemple de réponse suivant.

```
{
    "CustomKeyStoreId": cks-1234567890abcdef0
}
```

Si l'opération échoue, corrigez l'erreur indiquée par l'exception, puis réessayez. Pour obtenir de l'aide supplémentaire, consultez Dépannage d'un magasin de clés personnalisé.

Si vous essayez de créer un magasin de AWS CloudHSM clés avec toutes les mêmes valeurs de propriétés qu'un magasin de AWS CloudHSM clés déconnecté existant, AWS KMS cela ne crée pas de nouveau magasin de AWS CloudHSM clés et ne génère aucune exception ni n'affiche d'erreur. AWS KMS Reconnaît plutôt le doublon comme la conséquence probable d'une nouvelle tentative et renvoie l'ID du magasin de AWS CloudHSM clés existant.

Suivant : Pour utiliser le magasin de AWS CloudHSM clés, connectez-le à son AWS CloudHSM cluster.

Afficher un magasin AWS CloudHSM de clés

Vous pouvez consulter les AWS CloudHSM principaux magasins de chaque compte et région à l'aide de la AWS KMS console ou de l'DescribeCustomKeyStoresopération.

Utilisation de la AWS KMS console

Lorsque vous consultez les AWS CloudHSM principaux magasins du AWS Management Console, vous pouvez voir ce qui suit :

- Nom et ID du magasin de clés personnalisé
- L'ID du AWS CloudHSM cluster associé
- Le nombre de personnes HSMs dans le cluster
- État actuel de la connexion

Une valeur d'état de connexion (Status) de Disconnected indique que le magasin de clés personnalisé est nouveau et n'a jamais été connecté, ou qu'il a été intentionnellement déconnecté de son AWS CloudHSM cluster. Toutefois, si vos tentatives d'utilisation d'une clé KMS dans un magasin de clés personnalisé connecté échouent, cela peut indiquer un problème avec le magasin de clés

personnalisé ou son AWS CloudHSM cluster. Pour obtenir de l'aide, veuillez consulter Comment corriger les clés KMS défaillantes.

Pour consulter les AWS CloudHSM principaux magasins d'un compte et d'une région donnés, procédez comme suit.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), AWS CloudHSM key stores (Magasins de clés).

Pour personnaliser l'affichage, cliquez sur l'icône d'engrenage qui apparaît sous le bouton Créer un magasin de clés.

Utilisation de l' AWS KMS API

Pour consulter vos AWS CloudHSM principaux magasins, utilisez l'DescribeCustomKeyStoresopération. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Toutefois, vous pouvez utiliser le paramètre `CustomKeyStoreName` ou `CustomKeyStoreId` (mais pas les deux) pour limiter la sortie à un magasin de clés personnalisé en particulier. Pour les AWS CloudHSM banques de clés, la sortie comprend l'ID et le nom de la banque de clés personnalisée, le type de banque de clés personnalisé, l'ID du AWS CloudHSM cluster associé et l'état de la connexion. Si l'état de la connexion indique une erreur, la sortie inclut également un code d'erreur qui décrit la raison de l'erreur.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Par exemple, la commande suivante renvoie tous les magasins de clés personnalisées du compte et de la région. Vous pouvez utiliser les paramètres `Marker` et `Limit` pour parcourir les magasins de clés personnalisés de la sortie.

```
$ aws kms describe-custom-key-stores
```

L'exemple de commande suivant utilise le paramètre `CustomKeyStoreName` pour obtenir uniquement le magasin de clés personnalisé avec le nom convivial `ExampleCloudHSMKeyStore`.

Vous pouvez utiliser le paramètre `CustomKeyStoreName` ou le paramètre `CustomKeyStoreId` (mais pas les deux) dans chaque commande.

L'exemple de sortie suivant représente un magasin de AWS CloudHSM clés connecté à son AWS CloudHSM cluster.

> **Note**
>
> Le `CustomKeyStoreType` champ a été ajouté à la `DescribeCustomKeyStores` réponse pour distinguer les magasins de AWS CloudHSM clés des magasins de clés externes.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleCloudHSMKeyStore
{
   "CustomKeyStores": [
      {
         "CloudHsmClusterId": "cluster-1a23b4cdefg",
         "ConnectionState": "CONNECTED",
         "CreationDate": "1.499288695918E9",
         "CustomKeyStoreId": "cks-1234567890abcdef0",
         "CustomKeyStoreName": "ExampleCloudHSMKeyStore",
         "CustomKeyStoreType": "AWS_CLOUDHSM",
         "TrustAnchorCertificate": "<certificate appears here>"
      }
   ]
}
```

Un `ConnectionState` de `Disconnected` indique qu'un magasin de clés personnalisé n'a jamais été connecté ou qu'il a été intentionnellement déconnecté de son AWS CloudHSM cluster. Toutefois, si les tentatives d'utilisation d'une clé KMS dans un magasin de AWS CloudHSM clés connecté échouent, cela peut indiquer un problème lié au magasin de AWS CloudHSM clés ou à son AWS CloudHSM cluster. Pour obtenir de l'aide, veuillez consulter Comment corriger les clés KMS défaillantes.

Si `ConnectionState` a la valeur `FAILED`, la réponse `DescribeCustomKeyStores` inclut un élément `ConnectionErrorCode` qui explique la raison de l'erreur.

Par exemple, dans la sortie suivante, la valeur `INVALID_CREDENTIALS` indique que la connexion du magasin de clés personnalisé a échoué, car le mot de passe `kmsuser` n'est pas valide. Pour obtenir

de l'aide sur ce sujet et sur d'autres échecs de connexion, consultez Dépannage d'un magasin de clés personnalisé.

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
   "CustomKeyStores": [
      {
         "CloudHsmClusterId": "cluster-1a23b4cdefg",
         "ConnectionErrorCode": "INVALID_CREDENTIALS",
         "ConnectionState": "FAILED",
         "CustomKeyStoreId": "cks-1234567890abcdef0",
         "CustomKeyStoreName": "ExampleCloudHSMKeyStore",
         "CustomKeyStoreType": "AWS_CLOUDHSM",
         "CreationDate": "1.499288695918E9",
         "TrustAnchorCertificate": "<certificate appears here>"
      }
   ]
}
```

En savoir plus :

- Afficher les magasins de clés externes
- Identifiez les clés KMS dans les magasins de AWS CloudHSM clés
- Journalisation des appels d' AWS KMS API avec AWS CloudTrail

Modifier les paramètres du magasin AWS CloudHSM clé

Vous pouvez modifier les paramètres d'un magasin de AWS CloudHSM clés existant. Le magasin de clés personnalisé doit être déconnecté de son AWS CloudHSM cluster.

Pour modifier les paramètres du magasin de AWS CloudHSM clés, procédez comme suit :

1. Déconnectez le magasin de clés personnalisé de son cluster AWS CloudHSM .

   Lorsque le magasin de clés personnalisé est déconnecté, vous ne pouvez pas créer AWS KMS keys (clés KMS) dans le magasin de clés personnalisé et vous ne pouvez pas utiliser les clés KMS qu'il contient pour des opérations cryptographiques.
2. Modifiez un ou plusieurs des AWS CloudHSM principaux paramètres du magasin.

   Vous pouvez modifier les paramètres suivants d'un magasin de clés personnalisé :

Le nom convivial du magasin de clés personnalisé.

Entrez un nouveau nom convivial. Le nouveau nom doit être unique parmi tous les magasins de clés personnalisés de votre Compte AWS.

> **Important**
>
> N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

L'ID de cluster du AWS CloudHSM cluster associé.

Modifiez cette valeur pour remplacer le AWS CloudHSM cluster d'origine par un cluster associé. Vous pouvez utiliser cette fonctionnalité pour réparer un magasin de clés personnalisé si son AWS CloudHSM cluster est endommagé ou supprimé.

Spécifiez un AWS CloudHSM cluster qui partage un historique de sauvegarde avec le cluster d'origine et qui répond aux exigences d'association avec un magasin de clés personnalisé, dont deux actifs HSMs dans des zones de disponibilité différentes. Les clusters qui partagent un historique des sauvegardes historique ont le même certificat de cluster. Pour afficher le certificat de cluster d'un cluster, utilisez l'DescribeClustersopération. Vous ne pouvez pas utiliser la fonctionnalité de modification pour associer le magasin de clés personnalisé à un cluster AWS CloudHSM sans relation.

Mot de passe actuel de l'`kmsuser` utilisateur de chiffrement (CU).

AWS KMS Indique le mot de passe actuel du `kmsuser` CU du AWS CloudHSM cluster. Cette action ne modifie pas le mot de passe du `kmsuser` CU dans le AWS CloudHSM cluster.

Si vous modifiez le mot de passe de la `kmsuser` CU dans le AWS CloudHSM cluster, utilisez cette fonctionnalité pour indiquer AWS KMS le nouveau `kmsuser` mot de passe. Dans le cas contraire, AWS KMS peut pas se connecter au cluster et toutes les tentatives pour connecter le magasin de clés personnalisé au cluster échouent.

3. Reconnectez le magasin de clés personnalisé à son cluster AWS CloudHSM .

Modifiez les paramètres de votre magasin de clés

Vous pouvez modifier les paramètres de votre magasin de AWS CloudHSM clés dans la AWS KMS console ou en utilisant l'UpdateCustomKeyStoreopération.

Utilisation de la AWS KMS console

Lorsque vous modifiez un magasin de AWS CloudHSM clés, vous pouvez modifier n'importe laquelle des valeurs configurables.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), AWS CloudHSM key stores (Magasins de clés).
4. Choisissez la ligne du magasin de AWS CloudHSM clés que vous souhaitez modifier.

   Si la valeur de la colonne Status n'est pas Disconnected, vous devez déconnecter le magasin de clés personnalisé avant de pouvoir le modifier. (Dans le menu Key store actions (Actions de magasin de clés), choisissez Disconnect [Déconnecter].)

   Lorsqu'un magasin de AWS CloudHSM clés est déconnecté, vous pouvez gérer le magasin de AWS CloudHSM clés et ses clés KMS, mais vous ne pouvez pas créer ou utiliser de clés KMS dans le magasin de AWS CloudHSM clés.
5. À partir du menu Key store actions (Actions de magasin de clés), choisissez Edit (Modifier).
6. Effectuez une ou plusieurs des actions suivantes :

   - Entrez un nouveau nom convivial pour le magasin de clés personnalisé.
   - Entrez l'ID de cluster d'un AWS CloudHSM cluster associé.
   - Entrez le mot de passe actuel de l'utilisateur `kmsuser` crypté dans le AWS CloudHSM cluster associé.
7. Choisissez Save (Enregistrer).

   Quand la procédure est réussie, un message décrit les paramètres que vous avez modifiés. Si elle ne réussit pas, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez Dépannage d'un magasin de clés personnalisé.

8. Reconnectez le magasin de clés personnalisé.

   Pour utiliser le magasin de AWS CloudHSM clés, vous devez le reconnecter après l'avoir modifié. Vous pouvez laisser le magasin de clés AWS CloudHSM déconnecté. Toutefois, tant qu'il est déconnecté, vous ne pouvez pas créer de clés KMS dans le magasin de AWS CloudHSM clés ni utiliser les clés KMS du magasin de clés dans le AWS CloudHSM cadre d'opérations cryptographiques.

Utilisation de l' AWS KMS API

Pour modifier les propriétés d'un magasin de AWS CloudHSM clés, utilisez l'UpdateCustomKeyStoreopération. Vous pouvez modifier plusieurs propriétés d'un magasin de clés personnalisé dans la même commande. Si l'opération aboutit, AWS KMS renvoie une réponse HTTP 200 et un objet JSON sans propriétés. Pour vérifier que les modifications sont effectives, utilisez l'DescribeCustomKeyStoresopération.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Commencez par utiliser DisconnectCustomKeyStorepour déconnecter le magasin de clés personnalisé de son AWS CloudHSM cluster. Remplacez l'exemple d'ID de magasin de clés personnalisé, cks-1234567890abcdef0, par un ID réel.

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Le premier exemple utilise UpdateCustomKeyStorepour remplacer le nom convivial du magasin de AWS CloudHSM clés par`DevelopmentKeys`. La commande utilise le `CustomKeyStoreId` paramètre pour identifier le magasin de AWS CloudHSM clés et `CustomKeyStoreName` pour spécifier le nouveau nom du magasin de clés personnalisé.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --new-
custom-key-store-name DevelopmentKeys
```

L'exemple suivant remplace le cluster associé à une banque de AWS CloudHSM clés par une autre sauvegarde du même cluster. La commande utilise le `CustomKeyStoreId` paramètre pour identifier le magasin de AWS CloudHSM clés et le `CloudHsmClusterId` paramètre pour spécifier le nouvel ID de cluster.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --cloud-hsm-cluster-id cluster-1a23b4cdefg
```

L'exemple suivant indique AWS KMS que le mot de `kmsuser` passe actuel est`ExamplePassword`. La commande utilise le `CustomKeyStoreId` paramètre pour identifier le magasin de AWS CloudHSM clés et le `KeyStorePassword` paramètre pour spécifier le mot de passe actuel.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --key-store-password ExamplePassword
```

La commande finale reconnecte le magasin de AWS CloudHSM clés à son AWS CloudHSM cluster. Vous pouvez laisser le magasin de clés personnalisé à l'état déconnecté, mais vous devez le connecter avant de pouvoir créer des clés KMS ou d'utiliser les clés KMS existantes pour les opérations de chiffrement. Remplacez l'exemple d'ID de magasin de clés personnalisé par un ID réel.

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Connectez un magasin AWS CloudHSM de clés

Les nouveaux magasins de AWS CloudHSM clés ne sont pas connectés. Avant de pouvoir créer et utiliser AWS KMS keys dans votre magasin de AWS CloudHSM clés, vous devez le connecter au AWS CloudHSM cluster associé. Vous pouvez connecter et déconnecter votre magasin de AWS CloudHSM clés à tout moment et consulter son état de connexion.

Vous n'êtes pas obligé de connecter votre magasin de AWS CloudHSM clés. Vous pouvez laisser un magasin de AWS CloudHSM clés dans un état déconnecté indéfiniment et le connecter uniquement lorsque vous en avez besoin. Cependant, vous pouvez tester la connexion régulièrement pour vérifier que les paramètres sont corrects et que le magasin peut être connecté.

> **Note**
>
> AWS CloudHSM les magasins de clés ont un état de `DISCONNECTED` connexion uniquement lorsque le magasin de clés n'a jamais été connecté ou que vous le déconnectez explicitement. Si l'état de votre connexion au magasin de AWS CloudHSM clés est le même `CONNECTED` mais que vous ne parvenez pas à l'utiliser, assurez-vous que le AWS CloudHSM cluster associé est actif et qu'il en contient au moins un actif HSMs. Pour obtenir de l'aide

concernant les connexions ayant échoué, veuillez consulter the section called “Dépannage d'un magasin de clés personnalisé”.

Lorsque vous connectez un magasin de AWS CloudHSM clés, AWS KMS trouvez le AWS CloudHSM cluster associé, vous y connectez, vous connectez au AWS CloudHSM client en tant qu'utilisateur `kmsuser` cryptographique (CU), puis modifiez le `kmsuser` mot de passe. AWS KMS reste connecté au AWS CloudHSM client tant que le magasin de AWS CloudHSM clés est connecté.

Pour établir la connexion, AWS KMS crée un groupe de sécurité nommé `kms-<custom key store ID>` dans le cloud privé virtuel (VPC) du cluster. Le groupe de sécurité dispose d'une règle unique qui autorise le trafic entrant en provenance du groupe de sécurité du cluster. AWS KMS crée également une Elastic Network Interface (ENI) dans chaque zone de disponibilité du sous-réseau privé du cluster. AWS KMS ajoute le ENIs au groupe `kms-<cluster ID>` de sécurité et le groupe de sécurité du cluster. La description de chaque ENI est `KMS managed ENI for cluster <cluster-ID>`.

Le processus de connexion peut prendre un certain temps pour s'achever, jusqu'à 20 minutes.

Avant de connecter le magasin de AWS CloudHSM clés, vérifiez qu'il répond aux exigences.

- Son AWS CloudHSM cluster associé doit contenir au moins un HSM actif. Pour trouver le nombre de HSMs dans le cluster, visualisez le cluster dans la AWS CloudHSM console ou utilisez l'DescribeClustersopération. Si nécessaire, vous pouvez ajouter un module HSM.
- Le cluster doit disposer d'un compte utilisateur `kmsuser` crypté (CU), mais ce CU ne peut pas être connecté au cluster lorsque vous connectez le magasin de AWS CloudHSM clés. Pour obtenir de l'aide sur la déconnexion, reportez-vous à la section Comment se déconnecter et se reconnecter.
- L'état de connexion du magasin de AWS CloudHSM clés ne peut pas être `DISCONNECTING` ou`FAILED`. Pour afficher l'état de la connexion, utilisez la AWS KMS console ou la DescribeCustomKeyStoresréponse. Si l'état de la connexion est `FAILED`, déconnectez le magasin de clés personnalisé, résolvez le problème, puis connectez-le à nouveau.

Pour obtenir de l'aide concernant les connexions ayant échoué, veuillez consulter Comment corriger un échec de connexion.

Lorsque votre magasin de AWS CloudHSM clés est connecté, vous pouvez y créer des clés KMS et utiliser les clés KMS existantes dans des opérations cryptographiques.

Connectez-vous et reconnectez-vous à votre magasin de AWS CloudHSM clés

Vous pouvez connecter ou reconnecter votre magasin de AWS CloudHSM clés dans la AWS KMS console ou en utilisant l'ConnectCustomKeyStoreopération.

Utilisation de la AWS KMS console

Pour connecter un magasin de AWS CloudHSM clés dans le AWS Management Console, commencez par sélectionner le magasin de AWS CloudHSM clés sur la page Stockages de clés personnalisés. Le processus de connexion peut prendre jusqu'à 20 minutes.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), AWS CloudHSM key stores (Magasins de clés).
4. Choisissez la ligne du magasin de AWS CloudHSM clés que vous souhaitez connecter.

   Si l'état de connexion du magasin de AWS CloudHSM clés est Échec, vous devez déconnecter le magasin de clés personnalisé avant de le connecter.
5. Dans le menu Key store actions (Actions de magasin de clés), choisissez Connect (Connecter).

AWS KMS lance le processus de connexion à votre magasin de clés personnalisé. Il recherche le cluster AWS CloudHSM associé, crée l'infrastructure réseau requise, la connecte, se connecte au cluster AWS CloudHSM en tant qu'utilisateur de chiffrement (CU) `kmsuser` et effectue une rotation du mot de passe `kmsuser`. Une fois l'opération terminée, l'état de la connexion devient Connected.

Si l'opération échoue, un message d'erreur s'affiche qui décrit la raison de l'échec. Avant de réessayer de vous connecter, consultez l'état de connexion de votre magasin de AWS CloudHSM clés. Si le statut est Failed, vous devez déconnecter le magasin de clés personnalisé avant de vous connecter à nouveau. Si vous avez besoin d'aide, consultez Dépannage d'un magasin de clés personnalisé.

Suivant : the section called “Création d'une clé KMS dans un magasin de AWS CloudHSM clés”.

Utilisation de l' AWS KMS API

Pour connecter un magasin de AWS CloudHSM clés déconnecté, utilisez l'ConnectCustomKeyStoreopération. Le AWS CloudHSM cluster associé doit contenir au moins un HSM actif et l'état de connexion ne peut pas l'êtreFAILED.

Le processus de connexion peut prendre un certain temps pour s'achever, jusqu'à 20 minutes. Sauf si elle échoue rapidement, l'opération renvoie une réponse HTTP 200 et un objet JSON sans propriétés. Cependant, cette réponse initiale n'indique pas que la connexion a abouti. Pour déterminer l'état de connexion du magasin de clés personnalisé, consultez la DescribeCustomKeyStoresréponse.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Pour identifier le magasin de AWS CloudHSM clés, utilisez son identifiant de magasin de clés personnalisé. Vous pouvez trouver l'ID sur la page des stockages de clés personnalisés de la console ou en utilisant l'DescribeCustomKeyStoresopération sans paramètres. Avant d'exécuter cet exemple, remplacez l'ID de l'exemple par un ID valide.

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Pour vérifier que le magasin de AWS CloudHSM clés est connecté, utilisez l'DescribeCustomKeyStoresopération. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Toutefois, vous pouvez utiliser le paramètre `CustomKeyStoreName`ou `CustomKeyStoreId` (mais pas les deux) pour limiter la réponse à des magasins de clés personnalisés en particulier. Si `ConnectionState` a la valeur `CONNECTED`, cela indique que le magasin de clés personnalisé est connecté à son cluster AWS CloudHSM .

> **Note**
>
> Le `CustomKeyStoreType` champ a été ajouté à la `DescribeCustomKeyStores` réponse pour distinguer les magasins de AWS CloudHSM clés des magasins de clés externes.

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
   "CustomKeyStores": [
      "CustomKeyStoreId": "cks-1234567890abcdef0",
```

```
      "CustomKeyStoreName": "ExampleCloudHSMKeyStore",
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "CustomKeyStoreType": "AWS_CLOUDHSM",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "CONNECTED"
   ],
}
```

Si la valeur de `ConnectionState` est failed, l'élément `ConnectionErrorCode` indique la raison de l'échec. Dans ce cas, vous AWS KMS n'avez pas trouvé de AWS CloudHSM cluster dans votre compte avec l'ID du cluster`cluster-1a23b4cdefg`. Si vous avez supprimé le cluster, vous pouvez le restaurer à partir d'une sauvegarde du cluster d'origine, puis modifier l'ID de cluster pour le magasin de clés personnalisé. Pour obtenir de l'aide afin de répondre à un code d'erreur de connexion, veuillez consulter la rubrique Comment corriger un échec de connexion.

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
   "CustomKeyStores": [
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleKeyStore",
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "CustomKeyStoreType": "AWS_CLOUDHSM",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "FAILED"
      "ConnectionErrorCode": "CLUSTER_NOT_FOUND"
   ],
}
```

Déconnecter un magasin de AWS CloudHSM clés

Lorsque vous déconnectez un magasin de AWS CloudHSM clés, AWS KMS que vous vous déconnectez du AWS CloudHSM client, que vous vous déconnectez du AWS CloudHSM cluster associé et que vous supprimez l'infrastructure réseau créée pour prendre en charge la connexion.

Lorsqu'un magasin de AWS CloudHSM clés est déconnecté, vous pouvez gérer le magasin de AWS CloudHSM clés et ses clés KMS, mais vous ne pouvez pas créer ou utiliser de clés KMS dans le magasin de AWS CloudHSM clés. L'état de connexion du magasin de clés est `DISCONNECTED` et l'état de la clé des clés KMS du magasin de clés personnalisé est `Unavailable`, sauf si elles sont `PendingDeletion`. Vous pouvez reconnecter le magasin de AWS CloudHSM clés à tout moment.

> **Note**
>
> AWS CloudHSM les magasins de clés ont un état de `DISCONNECTED` connexion uniquement lorsque le magasin de clés n'a jamais été connecté ou que vous le déconnectez explicitement. Si l'état de votre connexion au magasin de AWS CloudHSM clés est le même `CONNECTED` mais que vous ne parvenez pas à l'utiliser, assurez-vous que le AWS CloudHSM cluster associé est actif et qu'il en contient au moins un actif HSMs. Pour obtenir de l'aide concernant les connexions ayant échoué, veuillez consulter the section called “Dépannage d'un magasin de clés personnalisé”.

Lorsque vous déconnectez un magasin de clés personnalisé, les clés KMS du magasin de clés deviennent immédiatement inutilisables (sous réserve d'une éventuelle cohérence). Toutefois, les ressources chiffrées à l'aide de clés de données protégées par la clé KMS ne sont pas affectées tant que la clé KMS n'est pas réutilisée, par exemple pour déchiffrer la clé de données. Ce problème affecte les Services AWS, dont beaucoup utilisent des clés de données pour protéger vos ressources. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

> **Note**
>
> Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

Pour mieux estimer l'effet de la déconnexion de votre magasin de clés personnalisé, identifiez les clés KMS du magasin de clés personnalisé et déterminez leur utilisation antérieure.

Vous pouvez déconnecter un magasin de AWS CloudHSM clés pour les raisons suivantes :

- Pour effectuer une rotation du mot de passe **`kmsuser`**. AWS KMS modifie le mode de passe de `kmsuser` chaque fois qu'il se connecte au cluster AWS CloudHSM . Pour forcer une rotation de mot de passe, déconnectez-vous et reconnectez-vous.
- Pour auditer le matériel clé des clés KMS du AWS CloudHSM cluster. Lorsque vous déconnectez le magasin de clés personnalisé AWS KMS , vous vous déconnectez du compte utilisateur `kmsuser` cryptographique du AWS CloudHSM client. Ceci vous permet de vous connecter au

cluster en tant qu'utilisateur du chiffrement `kmsuser`, et d'auditer et gérer les éléments de clé pour la clé KMS.

- Pour désactiver immédiatement toutes les clés KMS dans le magasin de clés AWS CloudHSM . Vous pouvez désactiver et réactiver les clés KMS dans un magasin de AWS CloudHSM clés en utilisant l'DisableKeyopération AWS Management Console ou. Ces opérations s'effectuent rapidement, mais elles agissent sur une seule clé KMS à la fois. La déconnexion du magasin de AWS CloudHSM clés change immédiatement l'état de toutes les clés KMS du magasin de AWS CloudHSM clés`Unavailable`, ce qui empêche leur utilisation dans le cadre d'une opération cryptographique.
- Pour réparer un échec de tentative de connexion. Si la tentative de connexion d'un magasin de AWS CloudHSM clés échoue (l'état de connexion du magasin de clés personnalisé est le cas`FAILED`), vous devez déconnecter le magasin de AWS CloudHSM clés avant de réessayer de le connecter.

Déconnectez votre magasin de AWS CloudHSM clés

Vous pouvez déconnecter votre AWS CloudHSM porte-clés dans la AWS KMS console ou en utilisant l'DisconnectCustomKeyStoreopération.

Déconnexion à l'aide de la AWS KMS console

Pour déconnecter un magasin de AWS CloudHSM clés connecté dans la AWS KMS console, commencez par choisir le magasin de AWS CloudHSM clés sur la page Stockages de clés personnalisés.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), AWS CloudHSM key stores (Magasins de clés).
4. Choisissez la ligne du magasin de clés externe que vous souhaitez déconnecter.
5. Dans le menu Key store actions (Actions de magasin de clés), choisissez Disconnect (Déconnecter).

Une fois l'opération terminée, l'état de la connexion passe de Disconnecting à Disconnected. Si l'opération échoue, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez Dépannage d'un magasin de clés personnalisé.

Déconnectez-vous à l'aide de l' AWS KMS API

Pour déconnecter un magasin de AWS CloudHSM clés connecté, utilisez l'DisconnectCustomKeyStoreopération. Si l'opération aboutit, AWS KMS renvoie une réponse HTTP 200 et un objet JSON sans propriétés.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Cet exemple déconnecte un magasin de AWS CloudHSM clés. Avant d'exécuter cet exemple, remplacez l'ID de l'exemple par un ID valide.

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Pour vérifier que le magasin de AWS CloudHSM clés est déconnecté, utilisez l'DescribeCustomKeyStoresopération. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Toutefois, vous pouvez utiliser le paramètre `CustomKeyStoreName` ou `CustomKeyStoreId` (mais pas les deux) pour limiter la réponse à des magasins de clés personnalisés en particulier. La `ConnectionState` valeur de `DISCONNECTED` indique que cet exemple de magasin de AWS CloudHSM clés n'est pas connecté à son AWS CloudHSM cluster.

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
   "CustomKeyStores": [
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionState": "DISCONNECTED",
      "CreationDate": "1.499288695918E9",
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleKeyStore",
      "CustomKeyStoreType": "AWS_CLOUDHSM",
      "TrustAnchorCertificate": "<certificate string appears here>"
   ],
}
```

Supprimer un magasin AWS CloudHSM de clés

Lorsque vous supprimez un magasin de AWS CloudHSM clés, toutes les AWS KMS métadonnées le AWS CloudHSM concernant sont supprimées de KMS, y compris les informations relatives à son association avec un AWS CloudHSM cluster. Cette opération n'affecte ni le AWS CloudHSM cluster HSMs, ni ses utilisateurs. Vous pouvez créer un nouveau magasin de AWS CloudHSM clés associé au même AWS CloudHSM cluster, mais vous ne pouvez pas annuler l'opération de suppression.

Vous ne pouvez supprimer qu'un magasin de AWS CloudHSM clés qui est déconnecté de son AWS CloudHSM cluster et qui n'en contient aucun AWS KMS keys. Avant de supprimer un magasin de clés personnalisé, procédez comme suit :

- Vérifiez que vous n'aurez jamais besoin d'utiliser l'une des clés KMS du magasin de clés pour des opérations de chiffrement. Ensuite, planifiez la suppression de toutes les clés KMS du magasin de clés. Pour obtenir de l'aide pour trouver les clés KMS dans un magasin de AWS CloudHSM clés, consultezTrouvez les clés KMS dans un magasin de AWS CloudHSM clés.
- Vérifiez que toutes les clés KMS ont été supprimées. Pour afficher les clés KMS dans un magasin de AWS CloudHSM clés, voirthe section called “Identifiez les clés KMS dans les magasins de AWS CloudHSM clés”.
- Déconnectez le magasin de AWS CloudHSM clés de son AWS CloudHSM cluster.

Au lieu de supprimer le magasin de AWS CloudHSM clés, pensez à le déconnecter de son AWS CloudHSM cluster associé. Lorsqu'un magasin de AWS CloudHSM clés est déconnecté, vous pouvez gérer le magasin de AWS CloudHSM clés et son AWS KMS keys. Mais vous ne pouvez pas créer ou utiliser de clés KMS dans le magasin de AWS CloudHSM clés. Vous pouvez reconnecter le magasin de AWS CloudHSM clés à tout moment.

Supprimer votre magasin AWS CloudHSM de clés

Vous pouvez supprimer votre magasin de AWS CloudHSM clés dans la AWS KMS console ou en utilisant cette DeleteCustomKeyStoreopération.

Utilisation de la AWS KMS console

Pour supprimer un magasin de AWS CloudHSM clés dans le AWS Management Console, commencez par sélectionner le magasin de AWS CloudHSM clés sur la page Stockages de clés personnalisés.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), AWS CloudHSM key stores (Magasins de clés).
4. Recherchez la ligne qui représente le magasin de AWS CloudHSM clés que vous souhaitez supprimer. Si l'état de connexion du magasin de AWS CloudHSM clés n'est pas Déconnecté, vous devez déconnecter le magasin de AWS CloudHSM clés avant de le supprimer.
5. Dans le menu Key store actions (Actions de magasin de clés), choisissez Delete (Supprimer).

Lorsque l'opération est terminée, un message de réussite apparaît et le magasin de AWS CloudHSM clés n'apparaît plus dans la liste des magasins de clés. Si l'opération échoue, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez Dépannage d'un magasin de clés personnalisé.

Utilisation de l' AWS KMS API

Pour supprimer un magasin de AWS CloudHSM clés, utilisez l'DeleteCustomKeyStoreopération. Si l'opération aboutit, AWS KMS renvoie une réponse HTTP 200 et un objet JSON sans propriétés.

Pour commencer, vérifiez que le magasin de AWS CloudHSM clés n'en contient aucune AWS KMS keys. Vous ne pouvez pas supprimer un magasin de clés personnalisé qui contient des clés KMS. Le premier exemple de commande utilise ListKeyset DescribeKeypour rechercher AWS KMS keys dans le magasin de AWS CloudHSM clés avec l'exemple d'ID de magasin de clés *cks-1234567890abcdef0* personnalisé. Dans ce cas, la commande ne renvoie aucune clé KMS. Si c'est le cas, utilisez l'ScheduleKeyDeletionopération pour planifier la suppression de chacune des clés KMS.

Bash

```
for key in $(aws kms list-keys --query 'Keys[*].KeyId' --output text) ;
do aws kms describe-key --key-id $key |
grep '"CustomKeyStoreId": "cks-1234567890abcdef0"' --context 100; done
```

PowerShell

```
PS C:\> Get-KMSKeyList | Get-KMSKey | where CustomKeyStoreId -eq
 'cks-1234567890abcdef0'
```

Ensuite, déconnectez le magasin de AWS CloudHSM clés. Cet exemple de commande utilise l'DisconnectCustomKeyStoreopération pour déconnecter un magasin de AWS CloudHSM clés de son AWS CloudHSM cluster. Avant d'exécuter la commande, remplacez l'exemple d'ID de magasin de clés personnalisé par un ID valide.

Bash

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

PowerShell

```
PS C:\> Disconnect-KMSCustomKeyStore -CustomKeyStoreId cks-1234567890abcdef0
```

Une fois le magasin de clés personnalisé déconnecté, vous pouvez utiliser DeleteCustomKeyStorecette opération pour le supprimer.

Bash

```
$ aws kms delete-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

PowerShell

```
PS C:\> Remove-KMSCustomKeyStore -CustomKeyStoreId cks-1234567890abcdef0
```

Dépannage d'un magasin de clés personnalisé

AWS CloudHSM les magasins clés sont conçus pour être disponibles et résilients. Cependant, vous devrez peut-être corriger certaines erreurs pour que votre magasin de AWS CloudHSM clés reste opérationnel.

Rubriques

- Comment corriger les clés KMS non disponibles
- Comment corriger les clés KMS défaillantes
- Comment corriger un échec de connexion
- Comment répondre à un échec d'opération de chiffrement
- Comment corriger les informations d'identification kmsuser non valides
- Comment supprimer les éléments de clé orphelins
- Comment récupérer les éléments de clé supprimés pour une clé KMS
- Comment se connecter en tant que kmsuser

Comment corriger les clés KMS non disponibles

L'état clé de AWS KMS keys dans un magasin de AWS CloudHSM clés est généralement`Enabled`. Comme toutes les clés KMS, l'état de la clé change lorsque vous désactivez les clés KMS dans un magasin de AWS CloudHSM clés ou que vous planifiez leur suppression. Toutefois, contrairement à d'autres clés KMS, les clés KMS d'un magasin de clés personnalisé peuvent également avoir un état de clé de `Unavailable`.

Un état de clé `Unavailable` indique que la clé KMS est dans un magasin de clés personnalisé qui a été intentionnellement déconnecté et que les tentatives pour le reconnecter, le cas échéant, ont échoué. Lorsqu'une clé KMS n'est pas disponible, vous pouvez afficher et gérer la clé KMS, mais vous ne pouvez pas l'utiliser dans les opérations de chiffrement.

Pour obtenir l'état de clé d'une clé KMS, sur la page Clés gérées par le client, veuillez consulter le champ Status (État) de la clé KMS. Vous pouvez également utiliser l'DescribeKeyopération et afficher l'`KeyState`élément dans la réponse. Pour plus de détails, veuillez consulter Identifier et afficher les clés.

Les clés KMS d'un magasin de clés personnalisé déconnecté possèdent l'état de clé `Unavailable` ou `PendingDeletion`. Les clés KMS dont la suppression a été planifiée à partir d'un magasin de clés personnalisé ont un état de clé `Pending Deletion`, même si le magasin de clés personnalisé est déconnecté. Cela vous permet d'annuler la suppression de clé planifiée sans reconnecter le magasin de clés personnalisé.

Pour corriger une clé KMS indisponible, reconnectez le magasin de clés personnalisé. Une fois le magasin de clés personnalisé reconnecté, l'état de clé des clés KMS du magasin de clés personnalisé est automatiquement restauré à son état précédent, comme `Enabled` ou `Disabled`. Les clés KMS qui sont en attente de suppression restent dans l'état `PendingDeletion`. Toutefois,

si le problème persiste, l'activation et la désactivation d'une clé KMS indisponible ne changent pas son état. L'action d'activation ou de désactivation prend effet uniquement lorsque la clé devient disponible.

Pour obtenir de l'aide concernant les connexions ayant échoué, consultez Comment corriger un échec de connexion.

Comment corriger les clés KMS défaillantes

Les problèmes liés à la création et à l'utilisation de clés KMS dans AWS CloudHSM les magasins de clés peuvent être dus à un problème lié à votre magasin de AWS CloudHSM clés, au AWS CloudHSM cluster associé, à la clé KMS ou à son contenu clé.

Lorsqu'un magasin de AWS CloudHSM clés est déconnecté de son AWS CloudHSM cluster, l'état clé des clés KMS dans le magasin de clés personnalisé est`Unavailable`. Toutes les demandes de création de clés KMS dans un magasin de AWS CloudHSM clés déconnecté renvoient une `CustomKeyStoreInvalidStateException` exception. Toutes les demandes pour chiffrer, déchiffrer, rechiffrer ou générer les clés de données renvoient une exception `KMSInvalidStateException`. Pour résoudre le problème, reconnectez le magasin de AWS CloudHSM clés.

Toutefois, vos tentatives d'utilisation d'une clé KMS dans un magasin de AWS CloudHSM clés pour des opérations cryptographiques peuvent échouer même si l'état de la clé est `Enabled` identique à celui de la connexion du magasin de AWS CloudHSM clés. `Connected` Cela peut être dû à l'une des conditions suivantes.

- Les éléments de clé de la clé KMS peuvent avoir été supprimés du cluster AWS CloudHSM associé. Pour étudier, recherchez l'identifiant de la clé d'une clé KMS et, si nécessaire, essayez de récupérer la clé.
- Tous HSMs ont été supprimés du AWS CloudHSM cluster associé au magasin de AWS CloudHSM clés. Pour utiliser une clé KMS dans un magasin de AWS CloudHSM clés dans le cadre d'une opération cryptographique, son AWS CloudHSM cluster doit contenir au moins un HSM actif. Pour vérifier le nombre et l'état de HSMs dans un AWS CloudHSM cluster, utilisez la AWS CloudHSM console ou l'DescribeClustersopération. Pour ajouter un HSM au cluster, utilisez la AWS CloudHSM console ou l'CreateHsmopération.
- Le AWS CloudHSM cluster associé au magasin de AWS CloudHSM clés a été supprimé. Pour corriger le problème, créez un cluster à partir d'une sauvegarde qui est liée au cluster d'origine, telle qu'une sauvegarde du cluster d'origine ou une sauvegarde qui a été utilisée pour créer

le cluster d'origine. Ensuite, modifiez l'ID de cluster dans les paramètres du magasin de clés personnalisé. Pour obtenir des instructions, veuillez consulter Comment récupérer les éléments de clé supprimés pour une clé KMS.

- Le AWS CloudHSM cluster associé au magasin de clés personnalisé ne disposait d'aucune session PKCS #11 disponible. Cela se produit généralement pendant les périodes de fort trafic en rafale, lorsque des sessions supplémentaires sont nécessaires pour traiter le trafic. Pour réagir à une exception `KMSInternalException` avec un message d'erreur concernant les sessions PKCS #11, revenez en arrière et relancez la requête.

Comment corriger un échec de connexion

Si vous essayez de connecter un magasin de AWS CloudHSM clés à son AWS CloudHSM cluster, mais que l'opération échoue, l'état de connexion du magasin de AWS CloudHSM clés passe à`FAILED`. Pour connaître l'état de connexion d'un magasin de AWS CloudHSM clés, utilisez la AWS KMS console ou l'DescribeCustomKeyStoresopération.

Sinon, certains tentatives de connexion échouent rapidement en raison d'erreurs de configuration de cluster facilement détectées. Dans ce cas, l'état de la connexion est toujours `DISCONNECTED`. Ces échecs renvoient un message d'erreur ou une exception qui explique pourquoi la tentative a échoué. Consultez la description de l'exception et les exigences du cluster, résolvez le problème, mettez à jour le magasin de AWS CloudHSM clés, si nécessaire, et réessayez de vous connecter.

Lorsque l'état de connexion est `FAILED` défini, exécutez l'DescribeCustomKeyStoresopération et voyez l'`ConnectionErrorCode`élément dans la réponse.

> **Note**
>
> Lorsque l'état de connexion d'un magasin de AWS CloudHSM clés est `FAILED` atteint, vous devez le AWS CloudHSM déconnecter avant de tenter de le reconnecter. Vous ne pouvez pas connecter un magasin de AWS CloudHSM clés doté d'un état de `FAILED` connexion.

- `CLUSTER_NOT_FOUND`indique qu'il est AWS KMS impossible de trouver un AWS CloudHSM cluster avec l'ID de cluster spécifié. Cela peut se produire parce que le mauvais ID de cluster a été fourni à une opération d'API ou que le cluster a été supprimé et n'a pas été remplacé. Pour corriger cette erreur, vérifiez l'ID du cluster, par exemple à l'aide de la AWS CloudHSM console ou de l'DescribeClustersopération. Si le cluster a été supprimé, la créez un cluster à partir d'une sauvegarde récente de l'original. Déconnectez ensuite le magasin de AWS CloudHSM clés,

modifiez le AWS CloudHSM paramètre d'ID du cluster de magasins de clés et reconnectez le magasin de AWS CloudHSM clés au cluster.

- `INSUFFICIENT_CLOUDHSM_HSM`indique que le AWS CloudHSM cluster associé n'en contient aucun HSMs. Pour vous connecter, le cluster doit avoir au moins un HSM. Pour trouver le nombre de HSMs dans le cluster, utilisez l'DescribeClustersopération. Pour résoudre cette erreur, ajoutez au moins un module HSM au cluster. Si vous en ajoutez plusieurs HSMs, il est préférable de les créer dans différentes zones de disponibilité.
- `INSUFFICIENT_FREE_ADDRESSES_IN_SUBNET`indique qu'il n' AWS KMS a pas pu connecter le magasin de AWS CloudHSM clés à son AWS CloudHSM cluster car au moins un sous-réseau privé associé au cluster ne dispose d'aucune adresse IP disponible. Une connexion au magasin de AWS CloudHSM clés nécessite une adresse IP libre dans chacun des sous-réseaux privés associés, bien que deux soient préférables.

  Vous ne pouvez pas ajouter des adresses IP (blocs CIDR) vers un sous-réseau existant. Si possible, déplacez ou supprimez les autres ressources qui utilisent les adresses IP du sous-réseau, telles que les EC2 instances inutilisées ou les interfaces réseau élastiques. Sinon, vous pouvez créer un cluster à partir d'une sauvegarde récente du AWS CloudHSM cluster avec des sous-réseaux privés nouveaux ou existants qui disposent de plus d'espace d'adressage libre. Ensuite, pour associer le nouveau cluster à votre magasin de AWS CloudHSM clés, déconnectez le magasin de clés personnalisé, remplacez l'ID de cluster du magasin de AWS CloudHSM clés par l'ID du nouveau cluster, puis réessayez de vous connecter.

  > **Tip**
  >
  > Pour éviter de réinitialiser le `kmsuser` mot de passe, utilisez la sauvegarde la plus récente du AWS CloudHSM cluster.

- `INTERNAL_ERROR`indique que la demande n' AWS KMS a pas pu être traitée en raison d'une erreur interne. Réitérez la demande. Pour les `ConnectCustomKeyStore` demandes, déconnectez le magasin de AWS CloudHSM clés avant de réessayer de vous connecter.
- `INVALID_CREDENTIALS`indique qu'il AWS KMS ne peut pas se connecter au AWS CloudHSM cluster associé car il ne possède pas le mot de passe de `kmsuser` compte correct. Pour obtenir de l'aide sur cette erreur, veuillez consulter Comment corriger les informations d'identification `kmsuser` non valides.
- `NETWORK_ERRORS` indique généralement des problèmes réseau temporaires. Déconnectez le magasin de AWS CloudHSM clés, attendez quelques minutes, puis réessayez de vous connecter.

- `SUBNET_NOT_FOUND`Indique qu'au moins un sous-réseau de la configuration du AWS CloudHSM cluster a été supprimé. Si vous AWS KMS ne trouvez pas tous les sous-réseaux dans la configuration du cluster, les tentatives de connexion du magasin de AWS CloudHSM clés au AWS CloudHSM cluster échouent.

  Pour corriger cette erreur, créez un cluster à partir d'une sauvegarde récente du même AWS CloudHSM cluster. (Ce processus crée une nouvelle configuration de cluster avec un VPC et des sous-réseaux privés.) Vérifiez que le nouveau cluster répond aux conditions requises pour un magasin de clés personnalisé et notez l'ID du nouveau cluster. Ensuite, pour associer le nouveau cluster à votre magasin de AWS CloudHSM clés, déconnectez le magasin de clés personnalisé, remplacez l'ID de cluster du magasin de AWS CloudHSM clés par l'ID du nouveau cluster, puis réessayez de vous connecter.

  > **Tip**
  >
  > Pour éviter de réinitialiser le `kmsuser` mot de passe, utilisez la sauvegarde la plus récente du AWS CloudHSM cluster.

- `USER_LOCKED_OUT` indique que le compte CU (utilisateur de chiffrement) `kmsuser` est verrouillé pour le cluster AWS CloudHSM associé en raison d'un trop grand nombre de tentatives de mot de passe ayant échoué. Pour obtenir de l'aide sur cette erreur, veuillez consulter Comment corriger les informations d'identification `kmsuser` non valides.

  Pour corriger cette erreur, déconnectez le magasin de AWS CloudHSM clés et utilisez la commande user change-password dans la CLI CloudHSM pour modifier le mot de passe du compte. `kmsuser` Ensuite, modifiez le `kmsuser` paramètre de mot de passe pour le magasin de clés personnalisé, et essayez de vous connecter à nouveau. Pour obtenir de l'aide, utilisez la procédure décrite dans la rubrique Comment corriger les informations d'identification `kmsuser` non valides.

- `USER_LOGGED_IN`indique que le compte `kmsuser` CU est connecté au AWS CloudHSM cluster associé. Cela AWS KMS empêche la rotation du mot de passe du `kmsuser` compte et la connexion au cluster. Pour corriger cette erreur, déconnectez le compte CU `kmsuser` du cluster. Si vous avez modifié le `kmsuser` mot de passe pour vous connecter au cluster, vous devez également mettre à jour la valeur du mot de passe du magasin de clés pour le magasin de AWS CloudHSM clés. Pour obtenir de l'aide, veuillez consulter Comment se déconnecter et se reconnecter.

- `USER_NOT_FOUND`indique qu'il est AWS KMS impossible de trouver un compte `kmsuser` CU dans le AWS CloudHSM cluster associé. Pour corriger cette erreur, créez un compte `kmsuser` CU dans le cluster, puis mettez à jour la valeur du mot de passe du magasin de clés pour le magasin de AWS CloudHSM clés. Pour obtenir de l'aide, veuillez consulter Comment corriger les informations d'identification `kmsuser` non valides.

Comment répondre à un échec d'opération de chiffrement

Une opération de chiffrement qui utilise une clé KMS dans un magasin de clés personnalisé peut échouer avec un `KMSInvalidStateException`. Les messages d'erreur suivants peuvent accompagner le `KMSInvalidStateException`.

KMS ne peut pas communiquer avec votre cluster CloudHSM. Il peut s'agir d'un problème réseau transitoire. Si cette erreur s'affiche à plusieurs reprises, vérifiez que les règles du réseau ACLs et du groupe de sécurité pour le VPC de votre AWS CloudHSM cluster sont correctes.

- Bien qu'il s'agisse d'une erreur HTTPS 400, elle peut résulter de problèmes réseau transitoires. Pour répondre, commencez par relancer la demande. Toutefois, si elle continue d'échouer, examinez la configuration de vos composants réseau. Cette erreur est probablement causée par une mauvaise configuration d'un composant réseau, telle qu'une règle de pare-feu ou une règle de groupe de sécurité VPC qui bloque le trafic sortant. Par exemple, KMS ne peut pas communiquer avec AWS CloudHSM les clusters IPv6. Pour plus de détails sur les prérequis, voirCréation d'un magasin de AWS CloudHSM clés.

KMS ne peut pas communiquer avec votre AWS CloudHSM cluster car l'utilisateur kmsuser est verrouillé. Si cette erreur s'affiche à plusieurs reprises, déconnectez le magasin de AWS CloudHSM clés et réinitialisez le mot de passe du compte kmsuser. Mettez à jour le mot de passe kmsuser pour le magasin de clés personnalisé et réessayez la demande.

- Ce message d'erreur indique que le compte CU (utilisateur de chiffrement) `kmsuser` est verrouillé pour le cluster AWS CloudHSM associé en raison d'un trop grand nombre de tentatives de mot de passe ayant échoué. Pour obtenir de l'aide sur cette erreur, veuillez consulter Comment se déconnecter et se connecter.

Comment corriger les informations d'identification **kmsuser** non valides

Lorsque vous connectez un magasin de AWS CloudHSM clés, vous AWS KMS vous connectez au AWS CloudHSM cluster associé en tant qu'utilisateur kmsuser cryptographique (CU). Il reste connecté jusqu'à ce que le magasin de AWS CloudHSM clés soit déconnecté. La réponse DescribeCustomKeyStores réponse affiche pour `ConnectionState` la valeur `FAILED` et pour `ConnectionErrorCode` la valeur `INVALID_CREDENTIALS`, comme indiqué dans l'exemple suivant.

Si vous déconnectez le magasin de AWS CloudHSM clés et modifiez le `kmsuser` mot de passe, vous AWS KMS ne pouvez pas vous connecter au AWS CloudHSM cluster avec les informations d'identification du compte `kmsuser` CU. Par conséquent, toutes les tentatives de connexion au magasin de AWS CloudHSM clés échouent. La réponse `DescribeCustomKeyStores` réponse affiche pour `ConnectionState` la valeur `FAILED` et pour `ConnectionErrorCode` la valeur `INVALID_CREDENTIALS`, comme indiqué dans l'exemple suivant.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleKeyStore
{
   "CustomKeyStores": [
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionErrorCode": "INVALID_CREDENTIALS"
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleKeyStore",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "FAILED"
   ],
}
```

De plus, au bout de cinq tentatives de connexion au cluster ayant échoué avec un mot de passe incorrect, AWS CloudHSM verrouille le compte utilisateur. Pour se connecter au cluster, vous devez modifier le mot de passe du compte.

S'il AWS KMS obtient une réponse de verrouillage lorsqu'il essaie de se connecter au cluster en tant que `kmsuser` CU, la demande de connexion au magasin de AWS CloudHSM clés échoue. La DescribeCustomKeyStoresréponse inclut un `ConnectionState` de `FAILED` et une `ConnectionErrorCode` valeur de`USER_LOCKED_OUT`, comme indiqué dans l'exemple suivant.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleKeyStore
```

```
{
   "CustomKeyStores": [
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionErrorCode": "USER_LOCKED_OUT"
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleKeyStore",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "FAILED"
   ],
}
```

Pour réparer l'une ou l'autre de ces conditions, utilisez la procédure suivante.

1. Déconnectez le magasin de AWS CloudHSM clés.
2. Exécutez l'DescribeCustomKeyStoresopération et visualisez la valeur de l'`ConnectionErrorCode`eélément dans la réponse.

   - Si la valeur de `ConnectionErrorCode` est `INVALID_CREDENTIALS`, déterminez le mot de passe actuel pour le compte `kmsuser`. Si nécessaire, utilisez la commande user change-password dans la CLI CloudHSM pour définir le mot de passe sur une valeur connue.
   - Si la `ConnectionErrorCode` valeur est`USER_LOCKED_OUT`, vous devez utiliser la commande user change-password dans la CLI CloudHSM pour modifier le mot de passe. `kmsuser`
3. Modifiez le mot de passe actuel de `kmsuser` afin qu'il corresponde au mot de passe actuel de `kmsuser` dans le cluster. Cette action indique à AWS KMS le mot de passe à utiliser pour se connecter au cluster. Elle ne change pas le mot de passe de `kmsuser` dans le cluster.
4. Connectez le magasin de clés personnalisé.

Comment supprimer les éléments de clé orphelins

Après avoir planifié la suppression d'une clé KMS d'un magasin de AWS CloudHSM clés, vous devrez peut-être supprimer manuellement le matériel clé correspondant du AWS CloudHSM cluster associé.

Lorsque vous créez une clé KMS dans un magasin de AWS CloudHSM clés, AWS KMS crée les métadonnées de la clé KMS dans le cluster associé AWS KMS et génère le matériel clé dans le AWS CloudHSM cluster associé. Lorsque vous planifiez la suppression d'une clé KMS dans un magasin de

AWS CloudHSM clés, les métadonnées de la clé KMS sont AWS KMS supprimées après la période d'attente. AWS KMS Fait ensuite de son mieux pour supprimer le matériel clé correspondant du AWS CloudHSM cluster. La tentative peut échouer si vous AWS KMS ne pouvez pas accéder au cluster, par exemple en cas de déconnexion du magasin de AWS CloudHSM clés ou de modification du `kmsuser` mot de passe. AWS KMS ne tente pas de supprimer les éléments clés des sauvegardes du cluster.

AWS KMS rapporte les résultats de sa tentative de suppression du contenu clé du cluster lors de `DeleteKey` la saisie de vos AWS CloudTrail journaux. Ils apparaissent dans l'élément `backingKeysDeletionStatus` de l'élément `additionalEventData`, comme illustré dans l'exemple d'entrée suivant. L'entrée inclut également l'ARN de la clé KMS, l'ID du AWS CloudHSM cluster et l'ID (`backing-key-id`) du matériau clé.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-12-10T14:23:51Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DeleteKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements":  {
        "keyId":"arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "additionalEventData": {
        "customKeyStoreId": "cks-1234567890abcdef0",
        "clusterId": "cluster-1a23b4cdefg",
        "backingKeys": "[{\"backingKeyId\":\"backing-key-id\"}]",
        "backingKeysDeletionStatus": "[{\"backingKeyId\":\"backing-key-id\",
\"deletionStatus\":\"FAILURE\"}]"
    },
    "eventID": "c21f1f47-f52b-4ffe-bff0-6d994403cf40",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
```

```
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-
west-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
    "managementEvent": true,
    "eventCategory": "Management"
}
```

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

Les procédures suivantes montrent comment supprimer le matériel clé orphelin du AWS CloudHSM cluster associé.

1. Déconnectez le magasin de AWS CloudHSM clés, s'il n'est pas déjà déconnecté, puis connectez-vous, comme expliqué dansComment se déconnecter et se connecter.

   > **Note**
   >
   > Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

2. Utilisez la commande key delete de la CLI CloudHSM pour supprimer la clé HSMs du cluster.

Toutes les entrées du CloudTrail journal relatives aux opérations cryptographiques effectuées avec une clé KMS dans un magasin de AWS CloudHSM clés incluent un `additionalEventData` champ avec le `customKeyStoreId` et`backingKey`. La valeur renvoyée dans le `backingKeyId` champ est l'attribut clé `id` CloudHSM. Nous vous recommandons de filtrer l'opération de suppression des clés `id` afin de supprimer le contenu clé orphelin que vous avez identifié dans vos CloudTrail journaux.

AWS CloudHSM reconnaît la `backingKeyId` valeur sous forme de valeur hexadécimale. Pour filtrer par`id`, vous devez ajouter le mot `backingKeyId` avec`0x`. Par exemple, si `backingKeyId` dans votre CloudTrail journal l'est`1a2b3c45678abcdef`, vous devez filtrer par**`0x`**`1a2b3c45678abcdef`.

L'exemple suivant supprime une clé de HSMs votre cluster. Le `backing-key-id` est répertorié dans l'entrée du CloudTrail journal. Avant d'exécuter cette commande, remplacez l'exemple `backing-key-id` par un exemple valide provenant de votre compte.

```
aws-cloudhsm key delete --filter attr.id="0x<backing-key-id>"
{
  "error_code": 0,
  "data": {
    "message": "Key deleted successfully"
  }
}
```

3. Déconnectez-vous et reconnectez le magasin de AWS CloudHSM clés comme décrit dansComment se déconnecter et se reconnecter.

Comment récupérer les éléments de clé supprimés pour une clé KMS

Si le contenu clé d'un AWS KMS key est supprimé, la clé KMS est inutilisable et tout le texte chiffré sous la clé KMS ne peut pas être déchiffré. Cela peut se produire si le matériel clé d'une clé KMS dans un magasin de AWS CloudHSM clés est supprimé du AWS CloudHSM cluster associé. Toutefois, il peut être possible de récupérer les clés.

Lorsque vous créez une AWS KMS key (clé KMS) dans un magasin de AWS CloudHSM clés AWS KMS , connectez-vous au AWS CloudHSM cluster associé et créez le matériel clé pour la clé KMS. Il remplace également le mot de passe par une valeur qu'il est le seul à connaître et reste connecté tant que le magasin de AWS CloudHSM clés est connecté. Étant donné que seul le propriétaire de

la clé, c'est-à-dire le CU qui a créé une clé, peut supprimer la clé, il est peu probable que la clé soit supprimée HSMs accidentellement.

Toutefois, si le contenu clé d'une clé KMS est supprimé HSMs d'un cluster, l'état de la clé KMS devient finalement`UNAVAILABLE`. Si vous essayez d'utiliser la clé KMS pour une opération cryptographique, l'opération échoue avec une exception `KMSInvalidStateException`. Et surtout, toutes les données chiffrées à l'aide de la clé KMS ne peuvent pas être déchiffrées.

Dans certains cas, vous pouvez récupérer les clés supprimés par en créant un cluster à partir d'une sauvegarde qui contient les clés. Cette stratégie fonctionne uniquement lorsqu'au moins une sauvegarde a été créée, tandis que la clé existait et avant qu'elle n'ait été supprimée.

Utilisez la procédure suivante pour récupérer les éléments de clé.

1. Recherchez une sauvegarde de cluster qui contient les éléments de clé. La sauvegarde doit également contenir tous les utilisateurs et toutes les clés dont vous avez besoin pour prendre en charge le cluster et ses données chiffrées.

   Utilisez l'DescribeBackupsopération pour répertorier les sauvegardes d'un cluster. Utilisez ensuite l'horodatage de la sauvegarde afin de vous aider à sélectionner une sauvegarde. Pour limiter la sortie au cluster associé au magasin de AWS CloudHSM clés, utilisez le `Filters` paramètre, comme indiqué dans l'exemple suivant.

   ```
   $ aws cloudhsmv2 describe-backups --filters clusterIds=<cluster ID>
   {
       "Backups": [
           {
               "ClusterId": "cluster-1a23b4cdefg",
               "BackupId": "backup-9g87f6edcba",
               "CreateTimestamp": 1536667238.328,
               "BackupState": "READY"
           },
                ...
       ]
   }
   ```

2. Créez un cluster à partir de la sauvegarde sélectionnée. Vérifiez que la sauvegarde contient la clé supprimée et les clés que le cluster nécessite.
3. Déconnectez le magasin de AWS CloudHSM clés afin de pouvoir modifier ses propriétés.

4. Modifiez l'ID de cluster du magasin de AWS CloudHSM clés. Entrez l'ID du cluster que vous avez créé à partir de la sauvegarde. Étant donné que le cluster partage un historique des sauvegardes avec le cluster d'origine, le nouvel ID de cluster doit être valide.
5. Reconnectez le magasin de AWS CloudHSM clés.

Comment se connecter en tant que **kmsuser**

Pour créer et gérer les éléments clés du AWS CloudHSM cluster pour votre magasin de AWS CloudHSM clés, utilisez AWS KMS le compte kmsuser Crypto User (CU). Vous créez le compte kmsuser CU dans votre cluster et vous fournissez son mot de passe AWS KMS lorsque vous créez votre magasin de AWS CloudHSM clés.

En général, AWS KMS gère le kmsuser compte. Toutefois, pour certaines tâches, vous devez déconnecter le magasin de AWS CloudHSM clés, vous connecter au cluster en tant que kmsuser CU et utiliser l'interface de ligne de commande (CLI) CloudHSM.

> **Note**
>
> Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

Cette rubrique explique comment déconnecter votre magasin de AWS CloudHSM clés et vous connecter en tant que telkmsuser, exécuter l'outil de ligne de AWS CloudHSM commande, puis vous déconnecter et reconnecter votre magasin de AWS CloudHSM clés.

Rubriques

- Comment se déconnecter et se connecter
- Comment se déconnecter et se reconnecter

Comment se déconnecter et se connecter

Utilisez la procédure suivante à chaque fois pour avoir besoin de vous connecter à un cluster associé en tant qu'utilisateur du kmsuser chiffrement.

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

1. Déconnectez le magasin de AWS CloudHSM clés, s'il ne l'est pas déjà. Vous pouvez utiliser la AWS KMS console ou AWS KMS l'API.

   Lorsque votre AWS CloudHSM clé est connectée, elle AWS KMS est connectée en tant que`kmsuser`. Cela vous empêche de vous connecter comme `kmsuser` ou de modifier le mot de passe `kmsuser`.

   Par exemple, cette commande permet DisconnectCustomKeyStorede déconnecter un exemple de magasin de clés. Remplacez l'exemple d'ID de magasin de AWS CloudHSM clés par un identifiant valide.

   ```
   $ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
   ```

2. Utilisez la commande de connexion pour vous connecter en tant qu'administrateur. Utilisez les procédures décrites dans la section Utilisation de la CLI CloudHSM du Guide AWS CloudHSM de l'utilisateur.

   ```
   aws-cloudhsm > login --username admin --role admin
             Enter password:
   {
     "error_code": 0,
     "data": {
       "username": "admin",
       "role": "admin"
     }
   }
   ```

3. Utilisez la commande user change-password de la CLI CloudHSM pour remplacer le mot de passe du compte par un mot de passe `kmsuser` que vous connaissez. (AWS KMS fait pivoter le mot de passe lorsque vous connectez votre magasin de AWS CloudHSM clés.) Le mot de passe doit contenir entre 7 et 32 caractères alphanumériques. Il est sensible à la casse et ne peut contenir aucun des caractères spéciaux.
4. Connectez-vous en `kmsuser` utilisant le mot de passe que vous avez défini. Pour obtenir des instructions détaillées, consultez la section Utilisation de la CLI CloudHSM du Guide AWS CloudHSM de l'utilisateur.

```
aws-cloudhsm > login --username kmsuser --role crypto-user
          Enter password:
{
  "error_code": 0,
  "data": {
    "username": "kmsuser",
    "role": "crypto-user"
  }
}
```

Comment se déconnecter et se reconnecter

Suivez la procédure suivante chaque fois que vous devez vous déconnecter en tant qu'utilisateur `kmsuser` cryptographique et reconnecter votre magasin de clés.

> **Remarques**
>
> Les procédures suivantes utilisent l'outil de ligne de commande du SDK AWS CloudHSM client 5, CloudHSM CLI. La CLI `key-handle` CloudHSM remplace par. `key-reference` Le 1er janvier 2025, la prise en charge des outils de ligne de commande du SDK client 3, de l'utilitaire de gestion CloudHSM (CMU) et de l'utilitaire de gestion des clés (KMU) AWS CloudHSM prendra fin. Pour plus d'informations sur les différences entre les outils de ligne de commande du SDK client 3 et l'outil de ligne de commande du SDK client 5, consultez la section Migrer de la CMU et de la KMU du SDK client 3 vers la CLI CloudHSM du SDK client 5 dans le guide de l'utilisateur.AWS CloudHSM

1. Effectuez la tâche, puis utilisez la commande logout dans la CLI CloudHSM pour vous déconnecter. Si vous ne vous déconnectez pas, les tentatives de reconnexion de votre magasin de AWS CloudHSM clés échoueront.

```
aws-cloudhsm  logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

2. Modifiez le paramètre de mot de passe de `kmsuser` pour le magasin de clés personnalisé.

   Cela indique AWS KMS le mot de passe actuel pour `kmsuser` le cluster. Si vous omettez cette étape, vous ne AWS KMS pourrez pas vous connecter au cluster et toutes les tentatives de reconnexion à votre banque de clés personnalisée échoueront. `kmsuser` Vous pouvez utiliser la AWS KMS console ou le `KeyStorePassword` paramètre de l'UpdateCustomKeyStoreopération.

   Par exemple, cette commande indique AWS KMS que le mot de passe actuel est`tempPassword`. Remplacez l'exemple de mot de passe par le mot de passe réel.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --
key-store-password tempPassword
```

3. Reconnectez le magasin de AWS KMS clés à son AWS CloudHSM cluster. Remplacez l'exemple d'ID de magasin de AWS CloudHSM clés par un identifiant valide. Au cours du processus de connexion, AWS KMS remplace le `kmsuser` mot de passe par une valeur qu'il est le seul à connaître.

   L'ConnectCustomKeyStoreopération revient rapidement, mais le processus de connexion peut prendre un certain temps. Cependant, cette réponse initiale n'indique pas que la connexion a réussi.

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

4. Utilisez cette DescribeCustomKeyStoresopération pour vérifier que le magasin de AWS CloudHSM clés est connecté. Remplacez l'exemple d'ID de magasin de AWS CloudHSM clés par un identifiant valide.

   Dans cet exemple, le champ d'état de connexion indique que le magasin de AWS CloudHSM clés est désormais connecté.

```
$ aws kms describe-custom-key-stores --custom-key-store-
id cks-1234567890abcdef0
{
   "CustomKeyStores": [
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleKeyStore",
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "CONNECTED"
   ],
}
```

Magasins de clés externes

Les magasins de clés externes vous permettent de protéger vos AWS ressources à l'aide de AWS clés cryptographiques externes. Cette fonctionnalité avancée est conçue pour les charges de travail réglementées que vous devez protéger avec des clés de chiffrement stockées dans un système de gestion des clés externe que vous contrôlez. Les magasins de clés externes soutiennent l'engagement de souveraineté AWS numérique qui vous donne le contrôle souverain de vos données AWS, y compris la possibilité de chiffrer avec des éléments clés que vous possédez et que vous contrôlez en dehors de AWS ceux-ci.

Un magasin de clés externe est un magasin de clés personnalisé soutenu par un gestionnaire de clés externe que vous possédez et gérez en dehors de celui-ci AWS. Votre gestionnaire de clés externe peut être un module de sécurité matériel physique ou virtuel (HSMs), ou tout système matériel ou logiciel capable de générer et d'utiliser des clés cryptographiques. Les opérations de chiffrement et de déchiffrement qui utilisent une clé KMS dans un magasin de clés externe sont effectuées par votre gestionnaire de clés externe à l'aide de votre clé cryptographique, une fonctionnalité connue sous le nom de « hold your own keys » ()HYOKs.

AWS KMS n'interagit jamais directement avec votre gestionnaire de clés externe et ne peut pas créer, afficher, gérer ou supprimer vos clés. Il AWS KMS interagit plutôt uniquement avec le logiciel proxy de stockage de clés externe (proxy XKS) que vous fournissez. Votre proxy de stockage de clés externe assure la médiation de toutes les communications entre AWS KMS et votre gestionnaire de clés externe. Il transmet toutes les demandes AWS KMS de votre gestionnaire de clés externe et retransmet les réponses de votre gestionnaire de clés externe à AWS KMS. Le proxy de stockage de clés externe traduit également les demandes génériques AWS KMS dans un format spécifique au

fournisseur que votre gestionnaire de clés externe peut comprendre, ce qui vous permet d'utiliser des magasins de clés externes avec des gestionnaires de clés de différents fournisseurs.

Vous pouvez utiliser des clés KMS dans un magasin de clés externe pour le chiffrement côté client, notamment avec l'AWS Encryption SDK. Mais les magasins de clés externes constituent une ressource importante pour le chiffrement côté serveur, car ils vous permettent de protéger vos AWS ressources de manière multiple Services AWS avec vos clés cryptographiques extérieures. AWS Services AWS qui prennent en charge les clés gérées par le client pour le chiffrement symétrique prennent également en charge les clés KMS dans un magasin de clés externe. Pour plus d'informations sur la prise en charge des services, consultez Intégration des services AWS.

Les magasins de clés externes vous permettent de les utiliser AWS KMS pour des charges de travail réglementées où les clés de chiffrement doivent être stockées et utilisées en dehors de AWS. Ils constituent toutefois une rupture majeure par rapport au modèle standard de responsabilité partagée et nécessitent des charges opérationnelles supplémentaires. Pour la plupart des clients, le risque accru pour la disponibilité et la latence dépassera les avantages en termes de sécurité perçus pour les magasins de clés externes.

Les magasins de clés externes vous permettent de contrôler la source de confiance. Les données chiffrées au moyen des clés KMS dans votre magasin de clés externe ne peuvent être déchiffrées qu'en utilisant le gestionnaire de clés externe que vous contrôlez. Si vous révoquez temporairement l'accès à votre gestionnaire de clés externe, par exemple en déconnectant le magasin de clés externe ou en déconnectant votre gestionnaire de clés externe du proxy de stockage de clés externe, AWS vous perdez tout accès à vos clés cryptographiques tant que vous ne les avez pas restaurées. Pendant cet intervalle, le texte chiffré qui a été chiffré au moyen de vos clés KMS ne peut pas être déchiffré. Si vous révoquez définitivement l'accès à votre gestionnaire de clés externe, tout le texte chiffré au moyen d'une clé KMS dans votre magasin de clés externe devient irrécupérable. Les seules exceptions sont les AWS services qui mettent brièvement en cache les clés de données protégées par vos clés KMS. Ces clés de données continuent de fonctionner jusqu'à ce que vous désactiviez la ressource ou que le cache expire. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

Les magasins de clés externes permettent de débloquer les quelques cas d'utilisation pour les charges de travail réglementées où les clés de chiffrement doivent rester sous votre contrôle exclusif et inaccessibles. AWS Mais il s'agit d'un changement majeur dans la façon dont vous exploitez une infrastructure basée sur le cloud et d'un changement significatif du modèle de responsabilité partagée. Pour la plupart des charges de travail, la charge opérationnelle supplémentaire et les

risques accrus en termes de disponibilité et de performances dépasseront les avantages en termes de sécurité perçus pour les magasins de clés externes.

Ai-je besoin d'un magasin de clés externe ?

Pour la plupart des utilisateurs, le magasin de AWS KMS clés par défaut, qui est protégé par des modules de sécurité matériels validés par la norme de sécurité FIPS 140-3 de niveau 3, répond à leurs exigences en matière de sécurité, de contrôle et de réglementation. Les utilisateurs de magasins de clés externes supportent des coûts, une maintenance et une charge de dépannage considérables, ainsi que des risques en matière de latence, de disponibilité et de fiabilité.

Lorsque vous envisagez un magasin de clés externe, prenez le temps de comprendre les alternatives, notamment un magasin de AWS CloudHSM clés soutenu par un AWS CloudHSM cluster que vous possédez et gérez, et des clés KMS contenant des éléments clés importés que vous générez vous-même HSMs et que vous pouvez supprimer des clés KMS à la demande. En particulier, l'importation d'éléments de clé ayant un délai d'expiration très court peut fournir un niveau de contrôle similaire sans risques pour la performance ou la disponibilité.

Un magasin de clés externe peut être la solution adaptée à votre organisation si vous répondez aux exigences suivantes :

- Vous devez utiliser des clés cryptographiques dans votre gestionnaire de clés local ou dans un gestionnaire de clés extérieur à AWS celui que vous contrôlez.
- Vous devez prouver que vos clés cryptographiques sont conservées uniquement sous votre contrôle en dehors du cloud.
- Vous devez pouvoir chiffrer et déchiffrer à l'aide de clés cryptographiques disposant d'une autorisation indépendante.
- Les clés doivent être soumises à un chemin d'audit indépendant secondaire.

Si vous choisissez un magasin de clés externe, limitez son utilisation aux charges de travail qui nécessitent une protection au moyen de clés cryptographiques en dehors d' AWS.

Modèle de responsabilité partagée

Les clés KMS standard utilisent des éléments clés générés et utilisés dans HSMs le cadre de la AWS KMS propriété et de la gestion. Vous établissez les politiques de contrôle d'accès sur vos clés

KMS et configurez Services AWS l'utilisation des clés KMS pour protéger vos ressources. AWS KMS assume la responsabilité de la sécurité, de la disponibilité, de la latence et de la durabilité du contenu clé de vos clés KMS.

Les clés KMS des magasins de clés externes dépendent des éléments et des opérations de clé de votre gestionnaire de clés externe. À ce titre, l'équilibre des responsabilités penche en votre faveur. Vous êtes responsable de la sécurité, de la fiabilité, de la durabilité et des performances des clés cryptographiques dans votre gestionnaire de clés externe. AWS KMS est chargé de répondre rapidement aux demandes et de communiquer avec le proxy de votre magasin de clés externe, ainsi que de maintenir nos normes de sécurité. Pour garantir que chaque clé externe stocke un texte chiffré au moins aussi solide que le texte AWS KMS chiffré standard, crypte d' AWS KMS abord tout le texte en clair avec des éléments clés spécifiques à votre AWS KMS clé KMS, puis l'envoie à votre gestionnaire de clés externe pour qu'il soit chiffré avec votre clé externe, une procédure connue sous le nom de double chiffrement. Par conséquent, ni AWS KMS ni le propriétaire des éléments de clé externes ne peuvent déchiffrer seuls le texte chiffré à double chiffrement.

Vous êtes responsable du maintien d'un gestionnaire de clés externe qui répond à vos normes réglementaires et de performance, de la fourniture et de la maintenance d'un proxy de magasin de clés externe conforme à la spécification de l'API du proxy de magasin de clés externe AWS KMS (langue française non garantie), ainsi que de la disponibilité et de la durabilité de vos éléments de clé. Vous devez également créer, configurer et maintenir un magasin de clés externe. Lorsque des erreurs sont causées par des composants que vous gérez, vous devez être prêt à les identifier et à les corriger afin que les AWS services puissent accéder à vos ressources sans interruption excessive. AWS KMS fournit des conseils de dépannage pour vous aider à déterminer la cause des problèmes et les solutions les plus probables.

Consultez les CloudWatch statistiques et les dimensions Amazon AWS KMS enregistrées pour les principaux magasins externes. AWS KMS recommande vivement de créer des CloudWatch alarmes pour surveiller votre magasin de clés externe afin de détecter les premiers signes de performances et de problèmes opérationnels avant qu'ils ne surviennent.

Qu'est-ce qui change ?

Les magasins de clés externes ne prennent en charge que les clés KMS de chiffrement symétriques. En interne AWS KMS, vous utilisez et gérez les clés KMS dans un magasin de clés externe de la même manière que vous gérez les autres clés gérées par les clients, notamment en définissant des politiques de contrôle d'accès et en surveillant l'utilisation des clés. Vous l'utilisez APIs avec les mêmes paramètres pour demander une opération cryptographique avec une clé KMS dans un magasin de clés externe que vous utilisez pour n'importe quelle clé KMS. La tarification est

également la même que pour les clés KMS standard. Pour plus de détails, voir Clés KMS dans des magasins de clés externes et AWS Key Management Service Tarification.

Toutefois, avec les magasins de clés externes, les principes suivants changent :

- Vous êtes responsable de la disponibilité, de la durabilité et de la latence des opérations de clé.
- Vous êtes responsable de tous les coûts liés au développement, à l'achat, à l'exploitation et à la licence de votre système de gestion de clés externe.
- Vous pouvez implémenter une autorisation indépendante pour toutes les demandes provenant AWS KMS de votre proxy de stockage de clés externe.
- Vous pouvez surveiller, auditer et consigner toutes les opérations de votre proxy de stockage de clés externe, ainsi que toutes les opérations de votre gestionnaire de clés externe liées aux AWS KMS demandes.

Où commencer ?

Pour créer et gérer un magasin de clés externe, vous devez choisir l'option de connectivité du proxy de votre magasin de clés externe, réunir les conditions préalables, puis créer et configurer votre magasin de clés externe.

Quotas

AWS KMS autorise jusqu'à 10 magasins de clés personnalisés dans chaque Compte AWS région, y compris les magasins de AWS CloudHSM clés et les magasins de clés externes, quel que soit leur état de connexion. En outre, il existe des quotas de requêtes AWS KMS concernant l'utilisation des clés KMS dans un magasin de clés externes.

Si vous choisissez la connectivité proxy VPC pour votre proxy de stockage de clés externe, des quotas peuvent également être appliqués aux composants requis VPCs, tels que les sous-réseaux et les équilibreurs de charge réseau. Pour plus d'informations sur ces quotas, consultez la console Service Quotas.

Régions

Pour minimiser la latence du réseau, créez les composants de votre magasin de clés externe dans la Région AWS la plus proche de votre gestionnaire de clés externe. Si possible, choisissez une région dont le temps d'aller-retour sur le réseau (RTT, round-trip time) est inférieur ou égal à 35 millisecondes.

Les magasins de clés externes sont pris en charge Régions AWS dans tous les pays pris en charge, à l'exception de la Chine (Pékin) et de la Chine (Ningxia). AWS KMS

Fonctions non prises en charge

AWS KMS ne prend pas en charge les fonctionnalités suivantes dans les magasins de clés personnalisés.

- Clés KMS asymétriques
- Clés KMS HMAC
- Clés KMS avec des éléments de clé importés
- Rotation automatique des clés
- Clés multi-région

En savoir plus :

- Announcing AWS KMS External Key Store (Annonce du magasin de clés externe) sur le Blog AWS News.

Concepts de magasins de clés externes

Apprenez les termes et concepts de base utilisés dans les magasins de clés externes.

Magasin de clés externe

Un magasin de clés externe est un magasin de clés AWS KMS personnalisé soutenu par un gestionnaire de clés externe autre AWS que celui que vous possédez et gérez. Chaque clé KMS d'un magasin de clés externe est associée à une clé externe dans votre gestionnaire de clés externe. Lorsque vous utilisez une clé KMS dans un magasin de clés externe à des fins de chiffrement ou de déchiffrement, l'opération est exécutée dans votre gestionnaire de clés externe à l'aide de votre clé externe, une disposition connue sous le nom de Hold your Own Keys (HYOK). Cette fonctionnalité est conçue pour les entreprises qui sont tenues de conserver leurs clés cryptographiques dans leur propre gestionnaire de clés externe.

Les magasins de clés externes garantissent que les clés cryptographiques et les opérations qui protègent vos AWS ressources restent dans votre gestionnaire de clés externe sous votre contrôle. AWS KMS envoie des demandes à votre gestionnaire de clés externe pour chiffrer et déchiffrer

les données, mais AWS KMS ne peut pas créer, supprimer ou gérer de clés externes. Toutes les demandes adressées AWS KMS à votre gestionnaire de clés externe sont traitées par un composant logiciel proxy de magasin de clés externe que vous fournissez, possédez et gérez.

AWS les services qui prennent en charge les clés gérées par le AWS KMS client peuvent utiliser les clés KMS de votre magasin de clés externe pour protéger vos données. En définitive, vos données sont bel et bien protégées par vos clés à l'aide de vos opérations de chiffrement dans votre gestionnaire de clés externe.

Les clés KMS d'un magasin de clés externe présentent des modèles de confiance, des accords de responsabilité partagée et des attentes en matière de performances fondamentalement différents de ceux des clés KMS standard. Avec les magasins de clés externes, vous êtes responsable de la sécurité et de l'intégrité des éléments de clé et des opérations cryptographiques. La disponibilité et la latence des clés KMS dans un magasin de clés externe sont affectées par le matériel, les logiciels, les composants réseau ainsi que par la distance entre AWS KMS et votre gestionnaire de clés externe. Vous êtes également susceptible d'encourir des coûts supplémentaires pour votre gestionnaire de clés externe et pour l'infrastructure de mise en réseau et d'équilibrage de charge avec laquelle votre gestionnaire de clés externe doit communiquer avec. AWS KMS

Vous pouvez utiliser votre magasin de clés externe dans le cadre de votre stratégie globale de protection des données. Pour chaque AWS ressource que vous protégez, vous pouvez décider laquelle nécessite une clé KMS dans un magasin de clés externe et laquelle peut être protégée par une clé KMS standard. Cela vous donne la possibilité de choisir des clés KMS pour des classifications de données, des applications ou des projets spécifiques.

Gestionnaire de clés externe

Un gestionnaire de clés externe est un composant extérieur à AWS qui peut générer des clés symétriques AES 256 bits et effectuer un chiffrement et un déchiffrement symétriques. Le gestionnaire de clés externe pour un magasin de clés externe peut être un module de sécurité matérielle (HSM) physique, un module HSM virtuel ou un gestionnaire de clés logiciel avec ou sans composant HSM. Il peut être situé n'importe où à l'extérieur AWS, y compris dans vos locaux, dans un centre de données local ou distant, ou dans n'importe quel cloud. Votre magasin de clés externe peut être soutenu par un seul gestionnaire de clés externe ou par plusieurs instances de gestionnaire de clés connexes qui partagent des clés cryptographiques, comme un cluster HSM. Les magasins de clés externes sont conçus pour prendre en charge divers gestionnaires externes provenant de différents fournisseurs. Pour plus d'informations sur la connexion à votre gestionnaire de clés externe, consultezChoisissez une option de connectivité proxy pour un magasin de clés externe.

Clé externe

Chaque clé KMS d'un magasin de clés externe est associée à une clé cryptographique dans votre gestionnaire de clés externe appelée clé externe. Lorsque vous chiffrez ou déchiffrez avec une clé KMS dans votre magasin de clés externe, l'opération cryptographique est effectuée dans votre gestionnaire de clés externe à l'aide de votre clé externe.

> **Warning**
>
> La clé externe est essentielle au fonctionnement de la clé KMS. En cas de perte ou de suppression de la clé externe, le texte chiffré au moyen de la clé KMS associée est irrécupérable.

Pour les magasins de clés externes, une clé externe doit être une clé AES 256 bits activée et capable d'effectuer le chiffrement et le déchiffrement. Pour obtenir des informations détaillées sur les exigences relatives aux clés externes, veuillez consulter la rubrique Exigences relatives à une clé KMS dans un magasin de clés externe.

AWS KMS Impossible de créer, de supprimer ou de gérer des clés externes. Les éléments de votre clé cryptographique ne quittent jamais votre gestionnaire de clés externe. Lorsque vous créez une clé KMS dans un magasin de clés externe, vous fournissez l'ID d'une clé externe (`XksKeyId`). Vous ne pouvez pas modifier l'ID de clé externe associé à une clé KMS, bien que votre gestionnaire de clés externe puisse effectuer une rotation des éléments de clé associés à l'ID de clé externe.

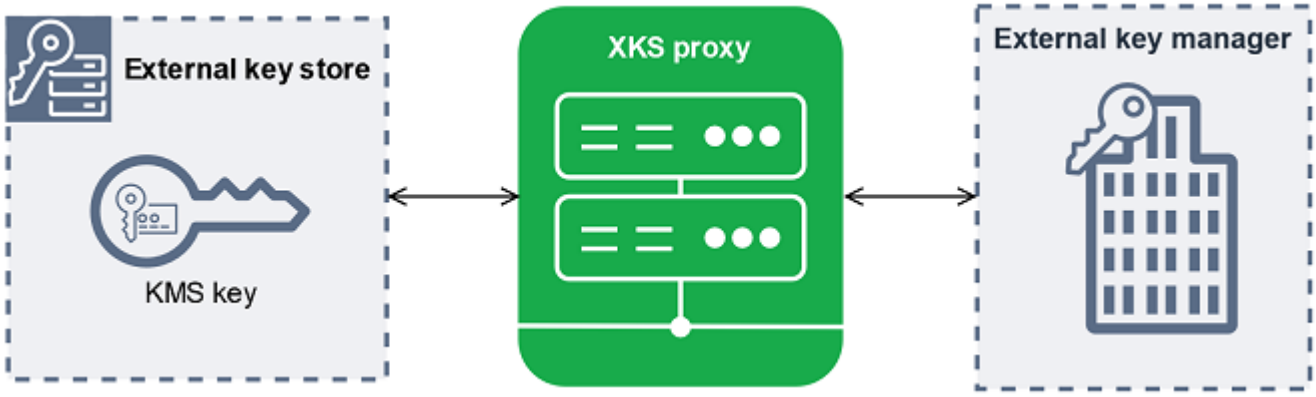
Outre votre clé externe, une clé KMS contenue dans un magasin de clés externe contient également des éléments de clé AWS KMS . Les données protégées par la clé KMS sont chiffrées d'abord AWS KMS à l'aide de la AWS KMS clé, puis par votre gestionnaire de clés externe à l'aide de votre clé externe. Ce processus de double chiffrement garantit que le texte chiffré protégé par votre clé KMS est toujours au moins aussi robuste que le texte chiffré protégé uniquement par AWS KMS.

De nombreuses clés cryptographiques possèdent différents types d'identifiants. Lorsque vous créez une clé KMS dans un magasin de clés externe, indiquez l'ID de la clé externe que le proxy de magasin de clés externe utilise pour faire référence à la clé externe. Si vous utilisez le mauvais identifiant, votre tentative de créer une clé KMS dans votre magasin de clés externe échoue.

Proxy de magasin de clés externe

Le proxy de stockage de clés externe (« proxy XKS ») est une application logicielle détenue et gérée par le client qui assure la médiation de toutes les communications entre AWS KMS et votre

gestionnaire de clés externe. Il traduit également les AWS KMS demandes génériques dans un format que votre gestionnaire de clés externe spécifique au fournisseur comprend. Un proxy de magasin de clés externe est requis pour un magasin de clés externe. Chaque magasin de clés externe est associé à exactement un proxy de magasin de clés externe.



AWS KMS Impossible de créer, de supprimer ou de gérer des clés externes. Vos éléments de clé cryptographique ne quittent jamais votre gestionnaire de clés externe. Toutes les communications entre AWS KMS et votre gestionnaire de clés externe sont assurées par le proxy de votre magasin de clés externe. AWS KMS envoie des demandes au proxy de stockage de clés externe et reçoit des réponses du proxy de stockage de clés externe. Le proxy de stockage de clés externe est chargé de transmettre les demandes AWS KMS à votre gestionnaire de clés externe et de transmettre les réponses de votre gestionnaire de clés externe à AWS KMS

Vous possédez et gérez le proxy de magasin de clés externe pour votre magasin de clés externe, et vous êtes responsable de sa maintenance et de son fonctionnement. Vous pouvez développer votre proxy de magasin de clés externe sur la base de la spécification d'API de proxy de magasin de clés externe open source qui AWS KMS publie ou achète une application proxy auprès d'un fournisseur. Votre proxy de magasin de clés externe est peut-être inclus dans votre gestionnaire de clés externe. Pour faciliter le développement de proxy, fournit AWS KMS également un exemple de proxy de stockage de clés externe (aws-kms-xks-proxy) et un client de test (xks-kms-xksproxy-test-client) qui vérifie que votre proxy de stockage de clés externe est conforme à la spécification.

Pour s'authentifier AWS KMS, le proxy utilise des certificats TLS côté serveur. Pour vous authentifier auprès de votre proxy, signez AWS KMS toutes les demandes adressées à votre proxy de stockage de clés externe avec un identifiant d'authentification du proxy SigV4. En option, votre proxy peut activer le protocole TLS mutuel (mTLS) pour avoir l'assurance supplémentaire qu'il n'accepte que les demandes provenant de. AWS KMS

Votre proxy de stockage de clés externe doit prendre en charge HTTP/1.1 ou version ultérieure et TLS 1.2 ou version ultérieure avec au moins l'une des suites de chiffrement suivantes :

- TLS_AES_256_GCM_ SHA384 (TLS 1.3)
- CHACHA2TLS_0_ POLY13 05_ SHA256 (TLS 1.3)

> **Note**
>
> Il AWS GovCloud (US) Region ne prend pas en charge TLS_ CHACHA2 POLY13 0_ 05_. SHA256

- SHA384 TLS_ECDHE_RSA_WITH_AES_256_GCM_ (TLS 1.2)
- SHA384 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_ (TLS 1.2)

Pour créer et utiliser les clés KMS dans votre magasin de clés externe, vous devez d'abord connecter le magasin de clés externe à son proxy de magasin de clés externe. Vous pouvez également déconnecter votre magasin de clés externe de son proxy à la demande. Dans ce cas, toutes les clés KMS du magasin de clés externe deviennent indisponibles ; elles ne peuvent être utilisées dans aucune opération cryptographique.

Connectivité du proxy de magasin de clés externe

La connectivité du proxy de stockage de clés externe (« connectivité du proxy XKS ») décrit la méthode AWS KMS utilisée pour communiquer avec votre proxy de magasin de clés externe.

Vous spécifiez votre option de connectivité de proxy lorsque vous créez votre magasin de clés externe, et elle devient une propriété du magasin de clés externe. Vous pouvez modifier l'option de connectivité de votre proxy en mettant à jour la propriété du magasin de clés personnalisé, mais vous devez vous assurer que votre proxy de magasin de clés externe peut toujours accéder aux mêmes clés externes.

AWS KMS prend en charge les options de connectivité suivantes :

- Connectivité des terminaux publics : AWS KMS envoie des demandes pour votre proxy de banque de clés externe via Internet à un point de terminaison public que vous contrôlez. Cette option est simple à créer et à gérer, mais elle peut ne pas répondre aux exigences de sécurité pour chaque installation.

- Connectivité du service de point de terminaison VPC : AWS KMS envoie des demandes à un service de point de terminaison Amazon Virtual Private Cloud (Amazon VPC) que vous créez et gérez. Vous pouvez héberger votre proxy de magasin de clés externe à l'intérieur de votre Amazon VPC, ou héberger votre proxy de magasin de clés externe à l'extérieur AWS et utiliser le VPC Amazon uniquement pour la communication.

Pour plus d'informations sur les options de connectivité du proxy de magasin de clés externe, veuillez consulter la rubrique Choisissez une option de connectivité proxy pour un magasin de clés externe.

Informations d'identification pour l'authentification du proxy de magasin de clés externe

Pour vous authentifier auprès de votre proxy de magasin de clés externe, AWS KMS signez toutes les demandes adressées à votre proxy de magasin de clés externe avec un identifiant d'authentification Signature V4 (SigV4). Vous établissez et maintenez les informations d'authentification sur votre proxy, puis vous les fournissez AWS KMS lorsque vous créez votre boutique externe.

> **Note**
>
> Les informations d'identification SigV4 AWS KMS utilisées pour signer les demandes adressées au proxy XKS n'ont aucun lien avec les informations d'identification SigV4 associées AWS Identity and Access Management aux principaux de votre compte. Comptes AWS Ne réutilisez aucune information d'identification IAM SigV4 pour votre proxy de magasin de clés externe.

Chaque information d'identification pour l'authentification du proxy comporte deux parties. Vous devez fournir les deux parties lors de la création d'un magasin de clés externe ou de la mise à jour des informations d'identification de l'authentification pour votre magasin de clés externe.

- ID de la clé d'accès : identifie la clé d'accès secrète. Vous pouvez fournir cet ID en texte brut.
- Clé d'accès secrète : partie secrète de l'identifiant. AWS KMS chiffre la clé d'accès secrète contenue dans les informations d'identification avant de les stocker.

Vous pouvez modifier le paramètre des informations d'identification à tout moment, par exemple lorsque vous saisissez des valeurs incorrectes, lorsque vous modifiez vos informations d'identification sur le proxy ou lorsque votre proxy effectue une rotation des informations d'identification. Pour obtenir

des informations techniques sur AWS KMS l'authentification auprès du proxy de stockage de clés externe, voir Authentification dans la spécification de l'API du proxy de stockage de clés AWS KMS externe.

Pour vous permettre de changer vos informations d'identification sans perturber les clés KMS Services AWS qui utilisent des clés KMS dans votre banque de clés externe, nous recommandons que le proxy de banque de clés externe prenne en charge au moins deux informations d'authentification valides pour. AWS KMS Cela garantit que vos anciennes informations d'identification continuent de fonctionner pendant que vous fournissez vos nouvelles informations d'identification à AWS KMS.

Pour vous aider à suivre l'âge de votre identifiant d'authentification proxy, définissez AWS KMS une CloudWatch métrique Amazon, XksProxyCredentialAge. Vous pouvez utiliser cette métrique pour créer une CloudWatch alarme qui vous avertit lorsque l'âge de votre identifiant atteint un seuil que vous avez établi.

Pour garantir que votre proxy de magasin de clés externe ne réponde qu'à AWS KMS, certains proxys de clés externes prennent en charge le protocole Transport Layer Security mutuel (mTLS). Pour plus de détails, consultez Authentification mTLS (facultatif).

Proxy APIs

Pour prendre en charge un magasin de clés AWS KMS externe, un proxy de magasin de clés externe doit implémenter le proxy requis, APIs comme décrit dans la spécification de l'API du proxy de stockage de clés AWS KMS externe. Ces demandes d'API proxy sont les seules requêtes AWS KMS envoyées au proxy. Bien que vous n'envoyiez jamais ces requêtes directement, le fait de les connaître peut vous aider à résoudre les problèmes qui pourraient survenir avec votre magasin de clés externe ou son proxy. Par exemple, AWS KMS inclut des informations sur la latence et les taux de réussite de ces appels d'API dans ses CloudWatch statistiques Amazon pour les magasins de clés externes. Pour plus de détails, consultez Surveillez les magasins de clés externes.

Le tableau suivant répertorie et décrit chacun des proxys APIs. Cela inclut également les AWS KMS opérations qui déclenchent un appel à l'API proxy et toutes les exceptions d' AWS KMS opération liées à l'API proxy.

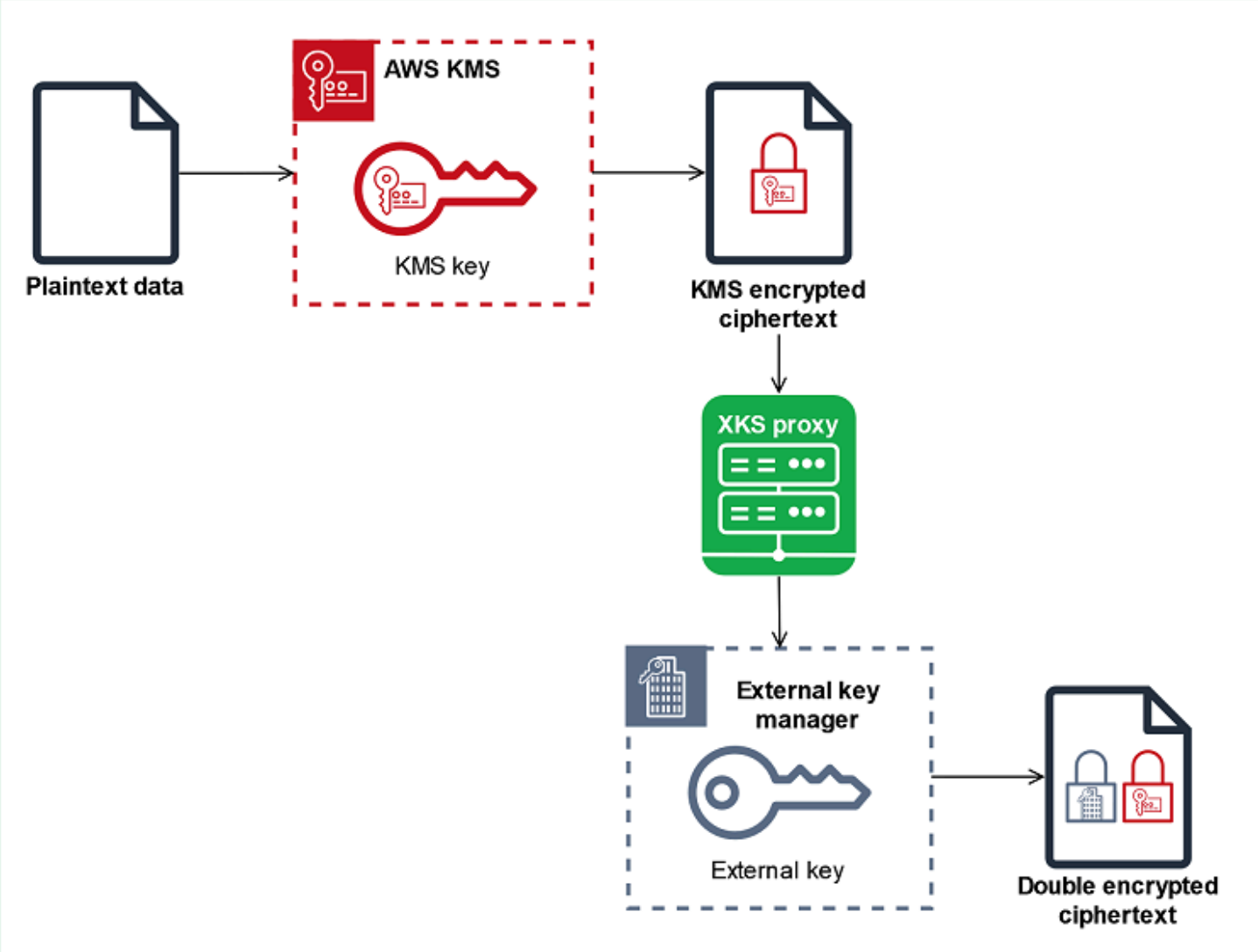
API de proxy	Description	AWS KMS Opérations associées
Decrypt	AWS KMS envoie le texte chiffré à déchiffrer, ainsi que l'ID de la clé	Déchiffrer, ReEncrypt

API de proxy	Description	AWS KMS Opérations associées
	externe à utiliser. L'algorithme de chiffrement requis est AES_GCM.	
Encrypt	AWS KMS envoie les données à chiffrer, ainsi que l'ID de la clé externe à utiliser. L'algorithme de chiffrement requis est AES_GCM.	Chiffrer, GenerateDataKey, GenerateDataKeyWithoutPlaintextReEncrypt
GetHealthStatus	AWS KMS demande des informations sur l'état du proxy et de votre gestionnaire de clés externe. L'état de chaque gestionnaire de clés externe peut être l'un des suivants. • `Active` : sain ; peut assurer le trafic • `Degraded` : défectueux, mais peut assurer le trafic • `Unavailable` : défectueux ; ne peut pas assurer le trafic	CreateCustomKeyStore(pour la connectivité des points de terminaison publics), ConnectCustomKeyStore(pour la connectivité des services de point de terminaison VPC) Si toutes les instances externes du gestionnaire de clés sont `Unavailable`, les tentatives de création ou de connexion du magasin de clés échouent avec l'exception `XksProxyUriUnreachableException`.
GetKeyMetadata	AWS KMS demande des informations sur la clé externe associée à une clé KMS dans votre banque de clés externe. La réponse inclut la spécification de la clé (`AES_256`), l'utilisation de la clé (`[ENCRYPT, DECRYPT]`) et indique si la clé externe est `ENABLED` ou `DISABLED`.	CreateKey Si la spécification de la clé n'est pas `AES_256`, si l'utilisation de la clé n'est pas `[ENCRYPT, DECRYPT]`, ou si l'état est `DISABLED`, l'opération `CreateKey` échoue avec l'exception `XksKeyInvalidConfigurationException`.

Double chiffrement

Les données chiffrées par une clé KMS dans un magasin de clés externe sont chiffrées deux fois. Tout d'abord, AWS KMS chiffre les données avec des AWS KMS éléments clés spécifiques à la clé KMS. Ensuite, le texte chiffré au moyen d' AWS KMS est chiffré par votre gestionnaire de clés externe à l'aide de votre clé externe. Ce processus est connu sous le nom de double chiffrement.

Le double chiffrement garantit que les données chiffrées par une clé KMS dans un magasin de clés externe sont au moins aussi robustes que le texte chiffré au moyen d'une clé KMS standard. Il protège également votre texte brut en transit depuis votre proxy AWS KMS de stockage de clés externe. Grâce au double chiffrement, vous gardez le contrôle total de vos textes chiffrés. Si vous révoquez définitivement l'accès d' AWS à votre clé externe par le biais de votre proxy externe, tout texte chiffré restant dans AWS est en fait détruit par chiffrement.



Pour activer le double chiffrement, chaque clé KMS d'un magasin de clés externe possède deux clés de sauvegarde cryptographiques :

- Matériau AWS KMS clé unique à la clé KMS. Ce matériel clé est généré et utilisé uniquement dans les modules de sécurité matériels certifiés AWS KMS FIPS 140-3 Security Level 3 (). HSMs
- Une clé externe dans votre gestionnaire de clés externe.

Le double chiffrement a les effets suivants :

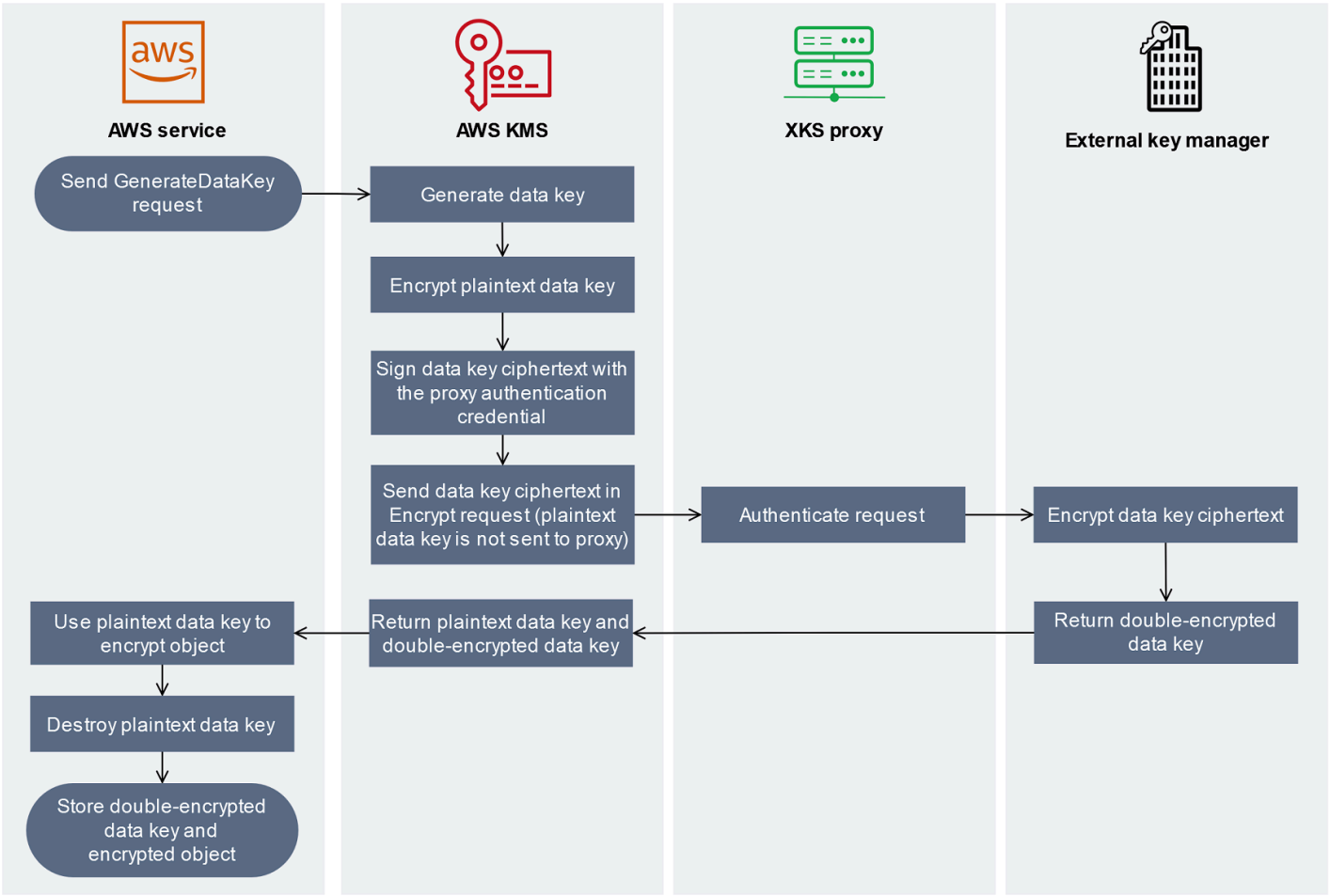
- AWS KMS ne peut déchiffrer aucun texte chiffré par une clé KMS dans un magasin de clés externe sans accéder à vos clés externes via votre proxy de stockage de clés externe.
- Vous ne pouvez pas déchiffrer un texte chiffré par une clé KMS dans un magasin de clés externe situé en dehors de celui-ci AWS, même si vous possédez le contenu de cette clé externe.
- Vous ne pouvez pas recréer une clé KMS qui a été supprimée d'un magasin de clés externe, même si vous possédez ses éléments de clé externes. Chaque clé KMS possède des métadonnées uniques qu'elle inclut dans le texte chiffré symétrique. Une nouvelle clé KMS ne serait pas en mesure de déchiffrer le texte chiffré au moyen de la clé d'origine, même si elle utilisait les mêmes éléments de clé externes.

Pour un exemple de double chiffrement en pratique, veuillez consulter la rubrique Fonctionnement des magasins de clés externes.

Fonctionnement des magasins de clés externes

Votre magasin de clés externe, votre proxy de magasin de clés externe et votre gestionnaire de clés externe travaillent ensemble pour protéger vos ressources AWS . La procédure suivante décrit le flux de travail de chiffrement d'un Service AWS typique qui chiffre chaque objet sous une clé de données unique protégée par une clé KMS. Dans ce cas, vous avez choisi une clé KMS dans un magasin de clés externe pour protéger l'objet. L'exemple montre comment AWS KMS utilise le double chiffrement pour protéger la clé de données en transit et garantir que le texte chiffré généré par une clé KMS dans un magasin de clés externe est toujours au moins aussi fort que le texte chiffré par une clé KMS symétrique standard contenant le contenu de la clé. AWS KMS

Les méthodes de cryptage utilisées par Service AWS chaque appareil intégré AWS KMS varient. Pour plus de détails, veuillez consulter la rubrique « Protection des données » dans le chapitre Sécurité de la documentation Service AWS .



1. Vous ajoutez un nouvel objet à votre Service AWS ressource. Pour chiffrer l'objet, Service AWS envoie une GenerateDataKeydemande à l' AWS KMS aide d'une clé KMS dans votre banque de clés externe.
2. AWS KMS génère une clé de données symétrique de 256 bits et prépare l'envoi d'une copie de la clé de données en texte clair à votre gestionnaire de clés externe via votre proxy de stockage de clés externe. AWS KMS lance le processus de double chiffrement en chiffrant la clé de données en texte brut avec le matériel AWS KMS clé associé à la clé KMS dans le magasin de clés externe.
3. AWS KMS envoie une demande de chiffrement au proxy de stockage de clés externe associé au magasin de clés externe. La demande inclut le texte chiffré de la clé de données à chiffrer et l'ID de la clé externe associée à la clé KMS. AWS KMS signe la demande en utilisant les informations d'authentification du proxy de votre magasin de clés externe.

   La copie en texte brut de la clé de données n'est pas envoyée au proxy du magasin de clés externes.

4. Le proxy de magasin de clés externe authentifie la requête, puis transmet la requête de chiffrement à votre gestionnaire de clés externe.

   Certains proxys de magasin de clés externe implémentent également une politique d'autorisation facultative qui permet uniquement à certains principaux d'effectuer des opérations dans des conditions spécifiques.

5. Votre gestionnaire de clés externe chiffre le texte chiffré de la clé de données à l'aide de la clé externe spécifiée. Le gestionnaire de clés externe renvoie la clé de données doublement chiffrée à votre proxy de magasin de clés externe, qui la renvoie à AWS KMS.
6. AWS KMS renvoie la clé de données en texte brut et la copie chiffrée deux fois de cette clé de données au. Service AWS
7. Il Service AWS utilise la clé de données en texte brut pour chiffrer l'objet de ressource, détruit la clé de données en texte clair et stocke la clé de données chiffrée avec l'objet chiffré.

   Certains Services AWS peuvent mettre en cache la clé de données en texte brut à utiliser pour plusieurs objets ou à réutiliser pendant que la ressource est en cours d'utilisation. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

Pour déchiffrer l'objet chiffré, vous Service AWS devez renvoyer la clé de données chiffrée AWS KMS dans une demande de déchiffrement. Pour déchiffrer la clé de données chiffrée, vous AWS KMS devez renvoyer la clé de données chiffrée à votre proxy de stockage de clés externe avec l'ID de la clé externe. Si la demande de déchiffrement adressée au proxy de stockage de clés externe échoue pour une raison quelconque, il est AWS KMS impossible de déchiffrer la clé de données cryptée et de déchiffrer l' Service AWS objet chiffré.

Contrôlez l'accès à votre magasin de clés externe

Toutes les fonctionnalités de contrôle d' AWS KMS accès (politiques clés, politiques IAM et autorisations) que vous utilisez avec les clés KMS standard fonctionnent de la même manière pour les clés KMS dans un magasin de clés externe. Vous pouvez utiliser les politiques IAM pour contrôler l'accès aux opérations d'API qui créent et gèrent des magasins de clés externes. Vous utilisez des politiques IAM et des politiques clés pour contrôler l'accès AWS KMS keys à votre banque de clés externe. Vous pouvez également utiliser des politiques de contrôle des services pour votre AWS organisation et des politiques de point de terminaison VPC pour contrôler l'accès aux clés KMS dans votre magasin de clés externe.

Nous vous recommandons de ne fournir aux utilisateurs et aux rôles que les autorisations dont ils ont besoin pour les tâches qu'ils sont susceptibles d'effectuer.

Rubriques

Autoriser des gestionnaires de magasins de clés externes

Les principaux qui créent et gèrent un magasin de clés externe doivent être autorisés pour les opérations de magasin de clés personnalisé. La liste suivante décrit les autorisations minimales requises pour les gestionnaires de magasin de clés externe. Étant donné qu'un magasin de clés personnalisé n'est pas une AWS ressource, vous ne pouvez pas accorder d'autorisation à un magasin de clés externe pour les principaux d'un autre Comptes AWS magasin.

- `kms:CreateCustomKeyStore`
- `kms:DescribeCustomKeyStores`
- `kms:ConnectCustomKeyStore`
- `kms:DisconnectCustomKeyStore`
- `kms:UpdateCustomKeyStore`
- `kms:DeleteCustomKeyStore`

Les principaux qui créent un magasin de clés externe doivent être autorisés à créer et à configurer les composants du magasin de clés externe. Les principaux ne peuvent créer des magasins de clés externes que sur leurs propres comptes. Pour créer un magasin de clés externe doté d'une connectivité au service de point de terminaison d'un VPC, les gestionnaires doivent être autorisés à créer les composants suivants :

- Un Amazon VPC
- Sous-réseaux publics et privés
- Un équilibreur de charge réseau et un groupe cible

- Un service de point de terminaison d'un Amazon VPC

Pour plus de détails, veuillez consulter les rubriques Identity and Access Management pour Amazon VPC, Identity and Access Management pour les points de terminaison de VPC et les services de points de terminaison de VPC et Elastic Load Balancing API permissions (Autorisations de l'API Elastic Load Balancing).

Autoriser les utilisateurs de clés KMS dans des magasins de clés externes

Les principaux responsables qui créent et gèrent AWS KMS keys dans votre magasin de clés externe ont besoin des mêmes autorisations que ceux qui créent et gèrent n'importe quelle clé KMS dans AWS KMS votre magasin de clés. La politique de clé par défaut pour les clés KMS d'un magasin de clés externe est identique à la politique de clé par défaut pour les clés KMS dans AWS KMS. Le contrôle d'accès par attributs (ABAC), qui utilise des balises et des alias pour contrôler l'accès aux clés KMS, fonctionne également sur les clés KMS dans les magasins de clés externes.

Les principaux qui utilisent les clés KMS dans votre magasin de clés personnalisé pour les opérations de chiffrement ont besoin des autorisations pour effectuer les opérations de chiffrement avec la clé KMS, telle que kms:Decrypt. Vous pouvez fournir ces autorisations dans une politique IAM ou une politique de clé. Cependant, les principaux n'ont pas besoin d'autorisations supplémentaires pour utiliser une clé KMS dans un magasin de clés personnalisé.

Pour définir une autorisation qui s'applique uniquement aux clés KMS d'un magasin de clés externe, utilisez la condition de politique `kms:KeyOrigin` avec la valeur de `EXTERNAL_KEY_STORE`. Vous pouvez utiliser cette condition pour limiter l'CreateKeyautorisation kms : ou toute autorisation spécifique à une ressource clé KMS. Par exemple, la politique IAM suivante permet à l'identité à laquelle elle est associée d'appeler les opérations spécifiées sur toutes les clés KMS du compte, à condition que les clés KMS se trouvent dans un magasin de clés externe. Notez que vous pouvez limiter l'autorisation aux clés KMS dans un magasin de clés externe et aux clés KMS dans un magasin de clés externe Compte AWS, mais pas à un magasin de clés externe spécifique du compte.

```
{
  "Sid": "AllowKeysInExternalKeyStores",
  "Effect": "Allow",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
```

```
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "EXTERNAL_KEY_STORE"
    }
  }
}
```

Autorisation AWS KMS de communication avec votre proxy de magasin de clés externe

AWS KMS communique avec votre gestionnaire de clés externe uniquement par le biais du proxy de stockage de clés externe que vous fournissez. AWS KMS s'authentifie auprès de votre proxy en signant ses demandes à l'aide du processus Signature Version 4 (SigV4) avec les informations d'identification d'authentification du proxy de stockage de clés externe que vous spécifiez. Si vous utilisez la connectivité d'un point de terminaison public pour votre proxy de magasin de clés externe, AWS KMS aucune autorisation supplémentaire n'est requise.

Toutefois, si vous utilisez la connectivité du service de point de terminaison VPC, vous devez AWS KMS autoriser la création d'un point de terminaison d'interface pour votre service de point de terminaison Amazon VPC. Cette autorisation est requise, que le proxy de banque de clés externe se trouve dans votre VPC ou qu'il se trouve ailleurs, mais qu'il utilise le service de point de terminaison du VPC pour communiquer avec. AWS KMS

AWS KMS Pour autoriser la création d'un point de terminaison d'interface, utilisez la console Amazon VPC ou l'ModifyVpcEndpointServicePermissionsopération. Accordez des autorisations au principal suivant : `cks.kms.<region>.amazonaws.com`.

Par exemple, la AWS CLI commande suivante permet de se AWS KMS connecter au service de point de terminaison VPC spécifié dans la région USA Ouest (Oregon) (us-west-2). Avant d'utiliser cette commande, remplacez l'identifiant du service Amazon VPC par Région AWS des valeurs valides pour votre configuration.

```
modify-vpc-endpoint-service-permissions
--service-id vpce-svc-12abc34567def0987
--add-allowed-principals '["cks.kms.us-west-2.amazonaws.com"]'
```

Pour supprimer cette autorisation, utilisez la console Amazon VPC ou ModifyVpcEndpointServicePermissionsavec le `RemoveAllowedPrincipals` paramètre.

Autorisation par proxy de magasin de clés externe (facultatif)

Certains proxys de magasin de clés externe mettent en œuvre des exigences d'autorisation pour l'utilisation de leurs clés externes. Un proxy de magasin de clés externe est autorisé, mais pas tenu, de concevoir et d'implémenter un schéma d'autorisation qui permet à des utilisateurs particuliers de demander des opérations particulières uniquement sous certaines conditions. Par exemple, un proxy peut être configuré pour permettre à l'utilisateur A de chiffrer avec une clé externe particulière, mais pas de déchiffrer à l'aide de cette clé.

L'autorisation du proxy est indépendante de l'authentification du proxy basée sur SIGv4 qui AWS KMS nécessite tous les proxys de stockage de clés externes. Elle est également indépendante des politiques de clés, des politiques IAM et des octrois qui accordent l'accès aux opérations affectant le magasin de clés externe ou ses clés KMS.

Pour activer l'autorisation par le proxy de stockage de clés externe, AWS KMS incluez des métadonnées dans chaque demande d'API proxy, y compris l'appelant, la clé KMS, l' AWS KMS opération, le Service AWS (le cas échéant). Les métadonnées de la requête pour la version 1 (v1) de l'API de proxy de clé externe se présentent comme suit.

```
"requestMetadata": {
    "awsPrincipalArn": string,
    "awsSourceVpc": string, // optional
    "awsSourceVpce": string, // optional
    "kmsKeyArn": string,
    "kmsOperation": string,
    "kmsRequestId": string,
    "kmsViaService": string // optional
}
```

Par exemple, vous pouvez configurer votre proxy pour autoriser les demandes d'un principal particulier (`awsPrincipalArn`), mais uniquement lorsque la demande est faite au nom du principal par un particulier Service AWS (`kmsViaService`).

Si l'autorisation du proxy échoue, l' AWS KMS opération correspondante échoue avec un message expliquant l'erreur. Pour plus de détails, veuillez consulter la rubrique Problèmes d'autorisation du proxy.

Authentification mTLS (facultatif)

Pour permettre à votre proxy de magasin de clés externe d'authentifier les demandes AWS KMS, AWS KMS signez toutes les demandes adressées à votre proxy de magasin de clés externe avec les informations d'identification d'authentification du proxy Signature V4 (SigV4) pour votre magasin de clés externe.

Pour garantir davantage que votre proxy de stockage de clés externe ne répond qu'aux AWS KMS demandes, certains proxys de clés externes prennent en charge le protocole MTL (Mutual Transport Layer Security), dans lequel les deux parties à une transaction utilisent des certificats pour s'authentifier mutuellement. Le mTLS ajoute l'authentification côté client, où le serveur proxy de stockage de clés externe authentifie le AWS KMS client, à l'authentification côté serveur fournie par le protocole TLS standard. Dans les rares cas où les informations d'identification d'authentification de votre proxy sont compromises, mTLS empêche une tierce partie d'effectuer des requêtes d'API au proxy de magasin de clés externe.

Pour implémenter le protocole mTLS, configurez votre proxy de magasin de clés externe de manière à n'accepter que les certificats TLS côté client présentant les propriétés suivantes :

- Le nom commun du sujet sur le certificat TLS doit être `cks.kms.<Region>.amazonaws.com`, par exemple, `cks.kms.eu-west-3.amazonaws.com`.
- Le certificat doit être lié à une autorité de certification associée à Amazon Trust Services.

Choisissez une option de connectivité proxy pour un magasin de clés externe

Avant de créer votre magasin de clés externe, choisissez l'option de connectivité qui détermine le mode de AWS KMS communication avec les composants de votre magasin de clés externe. L'option de connectivité que vous choisissez détermine le reste du processus de planification.

Si vous créez un magasin de clés externe, vous devez déterminer le mode de AWS KMS communication avec votre proxy de magasin de clés externe. Ce choix déterminera les composants dont vous avez besoin et la manière dont vous les configurez. AWS KMS prend en charge les options de connectivité suivantes. Choisissez l'option qui répond à vos objectifs de performance et de sécurité.

Avant de commencer, vérifiez que vous avez besoin d'un magasin de clés externe. La plupart des clients peuvent utiliser des clés KMS soutenues par du matériel AWS KMS clé.

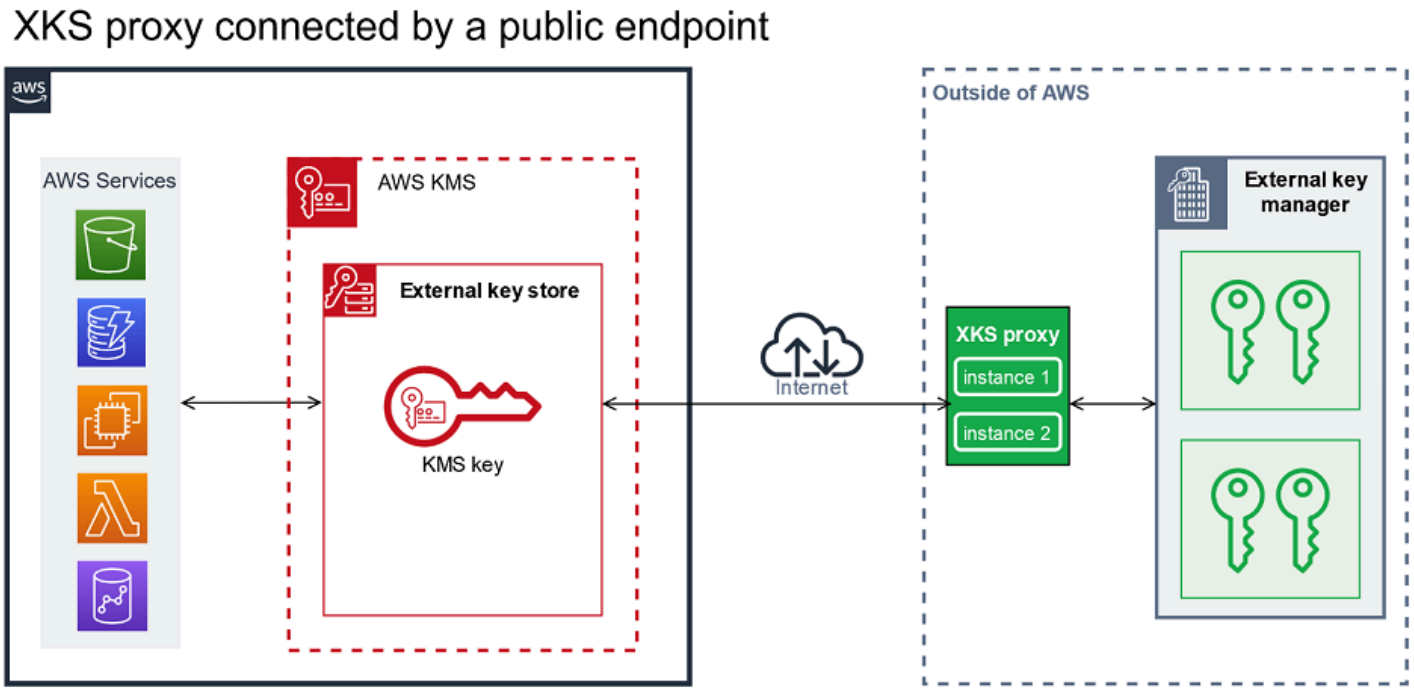
> **Note**
>
> Si votre proxy de magasin de clés externe est intégré à votre gestionnaire de clés externe, votre connectivité peut être prédéterminée. Pour obtenir des conseils, consultez la documentation de votre gestionnaire de clés externe ou de votre proxy de magasin de clés externe.

Vous pouvez modifier l'option de connectivité de votre proxy de magasin de clés externe, même sur un magasin de clés externe opérationnel. Toutefois, le processus doit être soigneusement planifié et exécuté afin de minimiser les interruptions, d'éviter les erreurs et de garantir un accès continu aux clés cryptographiques qui chiffrent vos données.

Connectivité au point de terminaison public

AWS KMS se connecte au proxy de stockage de clés externe (proxy XKS) via Internet à l'aide d'un point de terminaison public.

Cette option de connectivité est plus facile à configurer et à gérer, et elle s'adapte parfaitement à certains modèles de gestion des clés. Toutefois, il se peut qu'elle ne réponde pas aux exigences de sécurité de certaines organisations.



Prérequis

Si vous optez pour la connectivité au point de terminaison public, les éléments suivants sont requis.

- Le proxy de votre magasin de clés externe doit être accessible à partir d'un point de terminaison publiquement routable.
- Vous pouvez utiliser le même point de terminaison public pour plusieurs magasins de clés externes à condition qu'ils utilisent des valeurs de chemin d'URI de proxy différentes.
- Vous ne pouvez pas utiliser le même point de terminaison pour un magasin de clés externe connecté à un point de terminaison public et un magasin de clés externe doté d'une connectivité aux services de point de terminaison VPC Région AWS, même si les magasins de clés se trouvent dans des emplacements différents. Comptes AWS
- Vous devez obtenir un certificat TLS émis par une autorité de certification publique prise en charge pour les magasins de clés externes. Pour obtenir une liste, veuillez consulter les Autorités de certification approuvées (langue française non garantie).

  Le nom commun (CN) du sujet figurant sur le certificat TLS doit correspondre au nom de domaine indiqué dans le point de terminaison URI de proxy pour le proxy du magasin de clés externe. Par exemple, si le point de terminaison public est `https://myproxy.xks.example.com`, le TLS, le CN du certificat TLS doit être `myproxy.xks.example.com` ou `*.xks.example.com`.
- Assurez-vous que tous les pare-feux situés entre le proxy de stockage de clés externe AWS KMS et le proxy autorisent le trafic à destination et en provenance du port 443 du proxy. AWS KMS communique sur le port 443 IPv4. Cette valeur n'est pas configurable.

Pour connaître toutes les exigences relatives à un magasin de clés externe, veuillez consulter la rubrique Réunir les conditions préalables.

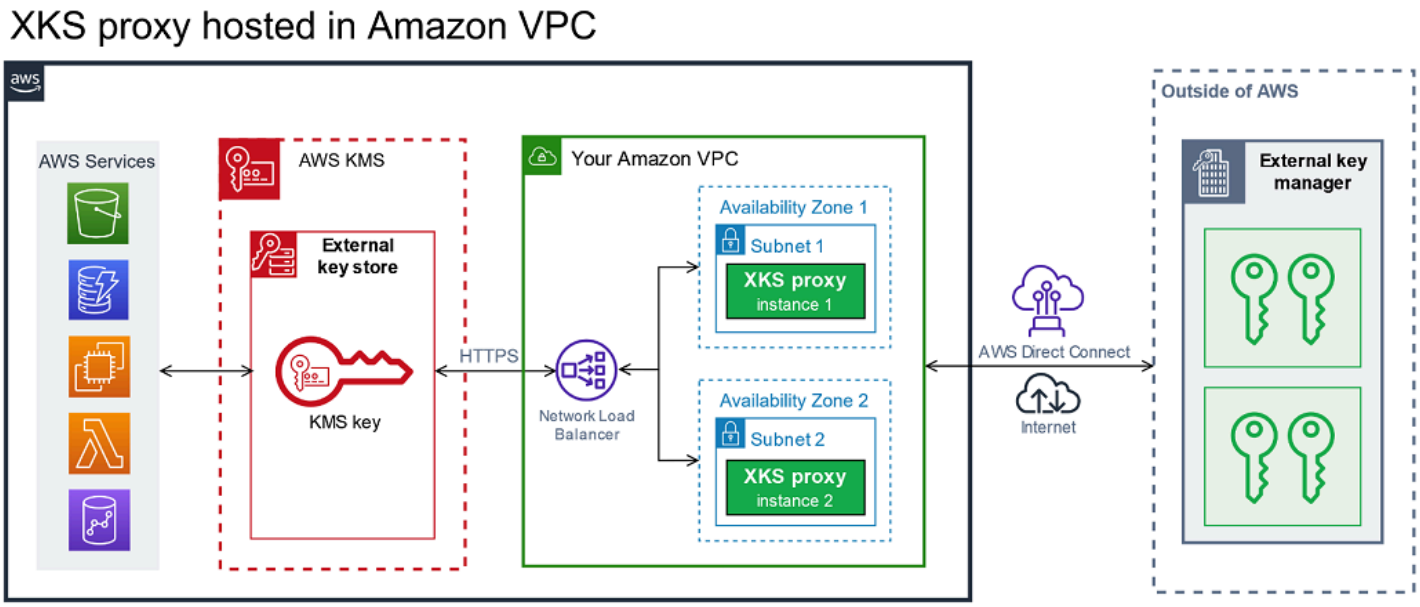
Connectivité au service de point de terminaison d'un VPC

AWS KMS se connecte au proxy de stockage de clés externe (proxy XKS) en créant un point de terminaison d'interface vers un service de point de terminaison Amazon VPC que vous créez et configurez. Vous êtes responsable de la création du service de point de terminaison d'un VPC et de la connexion de votre VPC à votre gestionnaire de clés externe.

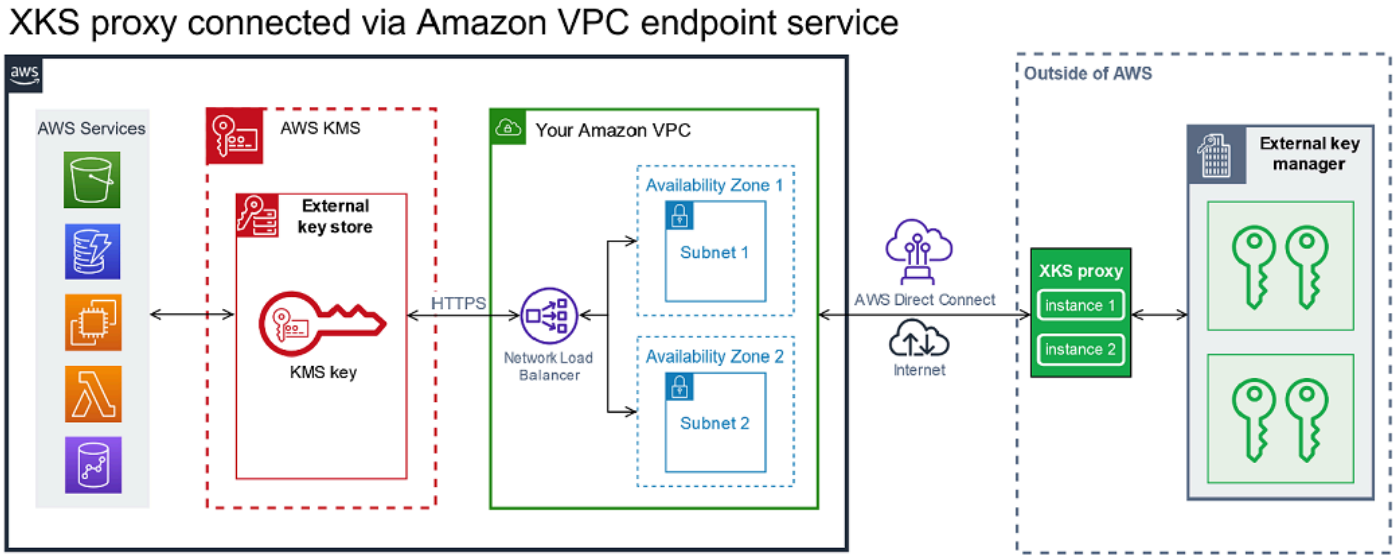
Votre service de point de terminaison peut utiliser n'importe laquelle des options network-to-Amazon VPC prises en charge pour les communications, notamment. AWS Direct Connect

Cette option de connectivité est plus compliquée à configurer et à gérer. Mais il utilise AWS PrivateLink, ce qui permet AWS KMS de se connecter en privé à votre Amazon VPC et à votre proxy de magasin de clés externe sans utiliser l'Internet public.

Vous pouvez localiser le proxy de votre magasin de clés externe dans votre Amazon VPC.



Vous pouvez également localiser votre proxy de stockage de clés externe à l'extérieur AWS et utiliser votre service de point de terminaison Amazon VPC uniquement pour une communication sécurisée avec. AWS KMS



En savoir plus :

- Passez en revue le processus de création d'un magasin de clés externe, ce qui inclut de réunir les conditions préalables. Cela vous aidera à vous assurer que vous disposez de tous les composants dont vous avez besoin lorsque vous créez votre magasin de clés externe.

- Découvrez comment contrôler l'accès à votre magasin de clés externe, notamment les autorisations requises par les administrateurs et les utilisateurs du magasin de clés externe.
- Découvrez les CloudWatch statistiques et les dimensions Amazon AWS KMS enregistrées pour les principaux magasins externes. Nous vous recommandons vivement de créer des alertes pour surveiller votre magasin de clés externe afin de détecter les premiers signes de problèmes de performance et de fonctionnement.

Configurer la connectivité du service de point de terminaison VPC

Suivez les instructions de cette section pour créer et configurer les AWS ressources et les composants associés requis pour un magasin de clés externe utilisant la connectivité du service de point de terminaison VPC. Les ressources répertoriées pour cette option de connectivité complètent les ressources requises pour tous les magasins de clés externes. Après avoir créé et configuré les ressources requises, vous pouvez créer votre magasin de clés externe.

Vous pouvez localiser votre proxy de stockage de clés externe dans votre Amazon VPC ou le localiser à l'extérieur AWS et utiliser votre service de point de terminaison VPC pour communiquer.

Avant de commencer, vérifiez que vous avez besoin d'un magasin de clés externe. La plupart des clients peuvent utiliser des clés KMS soutenues par du matériel AWS KMS clé.

> **Note**
>
> Certains des éléments requis pour la connectivité au service de point de terminaison d'un VPC peuvent être inclus dans votre gestionnaire de clés externe. En outre, votre logiciel peut avoir des exigences de configuration supplémentaires. Avant de créer et de configurer les AWS ressources de cette section, consultez la documentation de votre proxy et de votre gestionnaire de clés.

Rubriques

- Exigences pour la connectivité au service de point de terminaison d'un VPC
- Étape 1 : créer un Amazon VPC et des sous-réseaux
- Étape 2 : Création d'un groupe cible
- Étape 3 : créer un équilibreur de charge réseau
- Étape 4 : créer un service de point de terminaison VPC

- Étape 5 : Vérifiez votre nom de domaine DNS privé
- Étape 6 : Autoriser AWS KMS la connexion au service de point de terminaison VPC

Exigences pour la connectivité au service de point de terminaison d'un VPC

Si vous choisissez la connectivité au service de point de terminaison d'un VPC pour votre magasin de clés externe, les ressources suivantes sont requises.

Pour minimiser la latence du réseau, créez vos AWS composants dans le Région AWS support le plus proche de votre gestionnaire de clés externe. Si possible, choisissez une région dont le temps d'aller-retour sur le réseau (RTT, round-trip time) est inférieur ou égal à 35 millisecondes.

- Un Amazon VPC connecté à votre gestionnaire de clés externe. Il doit avoir au moins deux sous-réseaux privés dans deux zones de disponibilité différentes.

  Vous pouvez utiliser un Amazon VPC existant pour votre magasin de clés externe, à condition qu'il réponde aux exigences d'utilisation avec un magasin de clés externe. Plusieurs magasins de clés externes peuvent partager un Amazon VPC, mais chaque magasin de clés externe doit disposer de son propre service de point de terminaison d'un VPC et d'un nom DNS privé.
- Un service de point de terminaison d'un Amazon VPC à technologie AWS PrivateLink avec un équilibreur de charge réseau et un groupe cible.

  Le service de point de terminaison ne peut pas exiger d'acceptation. Vous devez également ajouter AWS KMS en tant que principal autorisé. Cela permet AWS KMS de créer des points de terminaison d'interface afin qu'elle puisse communiquer avec votre proxy de stockage de clés externe.
- Un nom DNS privé pour le service de point de terminaison d'un VPC qui est unique dans sa Région AWS.

  Le nom DNS privé doit être un sous-domaine d'un domaine public de niveau supérieur. Par exemple, si le nom DNS privé est `myproxy-private.xks.example.com`, il doit être un sous-domaine d'un domaine public tel que `xks.example.com` ou `example.com`.

  Vous devez vérifier la propriété du domaine DNS pour le nom DNS privé.
- Un certificat TLS émis par une autorité de certification publique prise en charge pour votre proxy de magasin de clés externe.

Le nom commun (CN) du sujet sur le certificat TLS doit correspondre au nom DNS privé. Par exemple, si le nom DNS privé est `myproxy-private.xks.example.com`, le CN du certificat TLS doit être `myproxy-private.xks.example.com` ou `*.xks.example.com`.

Pour connaître toutes les exigences relatives à un magasin de clés externe, veuillez consulter la rubrique Réunir les conditions préalables.

Étape 1 : créer un Amazon VPC et des sous-réseaux

La connectivité au service de point de terminaison d'un VPC nécessite un Amazon VPC connecté à votre gestionnaire de clés externe avec au moins deux sous-réseaux privés. Vous pouvez créer un Amazon VPC ou utiliser un Amazon VPC existant qui répond aux exigences relatives aux magasins de clés externes. Pour de plus amples informations sur la création d'un VPC, veuillez consulter la rubrique Créer un VPC dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Exigences pour votre Amazon VPC

Pour fonctionner avec des magasins de clés externes à l'aide de la connectivité au service de point de terminaison d'un VPC, l'Amazon VPC doit posséder les propriétés suivantes :

- Doit se trouver dans la même Compte AWS région prise en charge que votre magasin de clés externe.
- Comporter au moins deux sous-réseaux privés, chacun dans une zone de disponibilité différente.
- La plage d'adresses IP privées de votre Amazon VPC ne doit pas chevaucher la plage d'adresses IP privées du centre de données hébergeant votre gestionnaire de clés externe.
- Tous les composants doivent être utilisés IPv4.

Vous disposez de nombreuses options pour connecter l'Amazon VPC à votre proxy de magasin de clés externe. Choisissez une option qui répond à vos exigences de performance et de sécurité. Pour obtenir une liste, consultez Connecter votre VPC à d'autres réseaux et options de connectivité Network-to-Amazon VPC. Pour de plus amples informations, veuillez consulter AWS Direct Connect et le Guide de l'utilisateur AWS Site-to-Site VPN.

Créer un Amazon VPC pour votre magasin de clés externe

Utilisez les instructions suivantes pour créer l'Amazon VPC pour votre magasin de clés externe. Un Amazon VPC n'est requis que si vous choisissez l'option de connectivité au service de point de

terminaison d'un VPC. Vous pouvez utiliser un Amazon VPC existant qui répond aux exigences d'un magasin de clés externe.

Suivez les instructions de la section Créer un VPC, des sous-réseaux et d'autres ressources VPC à l'aide des valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
IPv4 Bloc d'adresse CIDR	Saisissez les adresses IP de votre VPC. La plage d'adresses IP privées de votre Amazon VPC ne doit pas chevaucher la plage d'adresses IP privées du centre de données hébergeant votre gestionnaire de clés externe.
Nombre de zones de disponibilité (AZs)	2 ou plus
Nombre de sous-réseaux publics	Pas d'exigence minimale (0)
Nombre de sous-réseaux privés	Un pour chaque AZ
Passerelles NAT	Pas d'exigence minimale.
Points de terminaison d'un VPC	Pas d'exigence minimale.
Enable DNS hostnames	Oui
Activer la résolution DNS	Oui

Assurez-vous de tester la communication de votre VPC. Par exemple, si votre proxy de magasin de clés externe ne se trouve pas dans votre Amazon VPC, créez une EC2 instance Amazon dans votre Amazon VPC, vérifiez que le VPC Amazon peut communiquer avec votre proxy de magasin de clés externe.

Connecter le VPC au gestionnaire de clés externe

Connectez le VPC au centre de données qui héberge votre gestionnaire de clés externe en utilisant l'une des options de connectivité réseau prises en charge par Amazon VPC. Assurez-vous que l' EC2 instance Amazon du VPC (ou le proxy de stockage de clés externe, s'il se trouve dans le VPC) peut communiquer avec le centre de données et le gestionnaire de clés externe.

Étape 2 : Création d'un groupe cible

Avant de créer le service de point de terminaison d'un VPC requis, créez ses composants requis, un équilibreur de charge réseau (NLB) et un groupe cible. L'équilibreur de charge réseau (NLB) distribue les requêtes entre plusieurs cibles saines, chacune pouvant répondre à la requête. Au cours de cette étape, vous créez un groupe cible avec au moins deux hôtes pour votre proxy de magasin de clés externe et vous enregistrez vos adresses IP auprès du groupe cible.

Suivez les instructions de la section Configure a target group (Configurer un groupe cible) à l'aide des valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Type de cible	Adresses IP
Protocole	TCP
Port	443
Type d'adresse IP	IPv4
VPC	Choisissez le VPC dans lequel vous allez créer le service de point de terminaison d'un VPC pour votre magasin de clés externe.
Protocole et chemin de	Votre protocole et votre chemin de surveillance de l'état varient en fonction de la configuration du proxy de votre magasin de clés externe. Consultez la

Champ	Valeur
surveillance de l'état	documentation de votre gestionnaire de clés externe ou de votre proxy de magasin de clés externe. Pour des informations générales sur la configuration de la surveillance de l'état de vos groupes cibles, veuillez consulter la rubrique Health checks for your target groups (Surveillance de l'état de vos groupes cibles) dans le Guide de l'utilisateur Elastic Load Balancing pour les Network Load Balancers.
Réseau	Autre adresse IP privée
IPv4 adresse	Les adresses privées de votre proxy de magasin de clés externe
Ports	443

Étape 3 : créer un équilibreur de charge réseau

L'équilibreur de charge réseau distribue le trafic réseau, y compris les requêtes d' AWS KMS auprès de votre proxy de magasin de clés externe, aux cibles configurées.

Suivez les instructions de la rubrique Configure a load balancer and a listener (Configurer un équilibreur de charge et un écouteur) pour configurer et ajouter un écouteur et créer un équilibreur de charge en utilisant les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Scheme	Internal (Interne)
Type d'adresse IP	IPv4
Mappage du réseau	Choisissez le VPC dans lequel vous allez créer le service de point de terminaison d'un VPC pour votre magasin de clés externe.
Mappage	Choisissez les deux zones de disponibilité (au moins deux) que vous avez configurées pour vos sous-réseaux VPC. Vérifiez les noms de sous-réseaux et l'adresse IP privée.

Champ	Valeur
Protocole	TCP
Port	443
Action par défaut : Réacheminer vers	Choisissez le groupe cible pour votre équilibreur de charge réseau.

Étape 4 : créer un service de point de terminaison VPC

En général, vous créez un point de terminaison vers un service. Toutefois, lorsque vous créez un service de point de terminaison VPC, vous êtes le fournisseur et vous AWS KMS créez un point de terminaison pour votre service. Pour un magasin de clés externe, créez un service de point de terminaison d'un VPC à l'aide de l'équilibreur de charge réseau que vous avez créé à l'étape précédente. Le service de point de terminaison VPC doit se trouver dans la même région prise Compte AWS en charge que votre magasin de clés externe.

Plusieurs magasins de clés externes peuvent partager un Amazon VPC, mais chaque magasin de clés externe doit disposer de son propre service de point de terminaison d'un VPC et d'un nom DNS privé.

Suivez les instructions de la rubrique Create an endpoint service (Créer un service de point de terminaison) pour créer votre service de point de terminaison d'un VPC avec les valeurs obligatoires suivantes. Pour les autres champs, acceptez les valeurs par défaut et fournissez les noms demandés.

Champ	Valeur
Nouveau type d'équilibreur de charge	Réseau
Équilibreurs de charge disponibles	Choisissez l'équilibreur de charge réseau que vous avez créé à l'étape précédente.

Champ	Valeur
	Si votre nouvel équilibreur de charge ne figure pas dans la liste, vérifiez que son état est actif. Il peut s'écouler quelques minutes avant que l'état de l'équilibreur de charge ne passe d'alloué à actif.
Acceptation requise	Faux. Désactivez la case à cocher. N'exigez pas d'acceptation. AWS KMS Impossible de se connecter au service de point de terminaison VPC sans acceptation manuelle. Si l'acceptation est requise, les tentatives de création du magasin de clés externe échouent avec une exception `XksProxyInvalidConfigurationException` .
Activer un nom DNS privé	Associez un nom DNS privé au service
Nom DNS privé	Saisissez un nom DNS privé unique dans sa Région AWS. Le nom DNS privé doit être un sous-domaine d'un domaine public de niveau supérieur. Par exemple, si le nom DNS privé est `myproxy-private.xks.example.com` , il doit être un sous-domaine d'un domaine public tel que `xks.example.com` ou `example.com` . Ce nom DNS privé doit correspondre au nom commun (CN) du sujet figurant dans le certificat TLS configuré sur le proxy de votre magasin de clés externe. Par exemple, si le nom DNS privé est `myproxy-private.xks.example.com` , le CN du certificat TLS doit être `myproxy-private.xks.example.com` ou `*.xks.example.com` . Si le certificat et le nom DNS privé ne correspondent pas, les tentatives de connexion d'un magasin de clés externe à son proxy de magasin de clés externe échouent avec le code d'erreur de connexion de `XKS_PROXY_INVALID_TLS_CONFIGURATION` . Pour en savoir plus, consultez Erreurs de configuration générale.
Types d'adresses IP pris en charge	IPv4

Étape 5 : Vérifiez votre nom de domaine DNS privé

Lorsque vous créez votre service de point de terminaison d'un VPC, son statut de vérification de domaine est `pendingVerification`. Avant d'utiliser le service de point de terminaison d'un VPC pour créer un magasin de clés externe, ce statut doit être `verified`. Pour vérifier que vous êtes bien le propriétaire du domaine associé à votre nom DNS privé, vous devez créer un enregistrement TXT sur un serveur DNS public.

Par exemple, si le nom DNS privé de votre service de point de terminaison VPC est`myproxy-private.xks.example.com`, vous devez créer un enregistrement TXT dans un domaine public, tel que `xks.example.com` ou`example.com`, selon ce qui est public. AWS PrivateLink recherche d'abord l'enregistrement TXT activé, `xks.example.com` puis activé`example.com`.

> **Tip**
>
> Après avoir ajouté un enregistrement TXT, il peut s'écouler quelques minutes avant que la valeur du Domain verification status (Statut de vérification du domaine) passe de `pendingVerification` à `verify`.

Pour commencer, identifiez le statut de vérification de votre domaine à l'aide de l'une des méthodes suivantes. Les valeurs valides sont `verified`, `pendingVerification` et `failed`.

- Dans la console Amazon VPC, choisissez Endpoint services (Services de points de terminaison), puis choisissez votre service de point de terminaison. Dans le volet d'informations, veuillez consulter la rubrique Domain verification status (Statut de vérification du domaine).
- Utilisez l'DescribeVpcEndpointServiceConfigurationsopération. La valeur de `State` se trouve dans le champ `ServiceConfigurations.PrivateDnsNameConfiguration.State`.

Si le statut de vérification n'est pas `verified`, suivez les instructions de la rubrique Domain ownership verification (Vérification de la propriété du domaine) pour ajouter un enregistrement TXT au serveur DNS de votre domaine et vérifier que l'enregistrement TXT est publié. Vérifiez ensuite à nouveau votre statut de vérification.

Vous n'êtes pas obligé de créer un enregistrement A pour le nom de domaine DNS privé. Lorsque vous AWS KMS créez un point de terminaison d'interface pour le service de point de terminaison de votre VPC, il crée AWS PrivateLink automatiquement une zone hébergée avec l'enregistrement A requis pour le nom de domaine privé dans le AWS KMS VPC. Pour les magasins de clés externes

dotés d'une connectivité au service de point de terminaison d'un VPC, cela se produit lorsque vous connectez votre magasin de clés externe à son proxy de magasin de clés externe.

Étape 6 : Autoriser AWS KMS la connexion au service de point de terminaison VPC

Vous devez l'ajouter AWS KMS à la liste des principaux autorisés pour votre service de point de terminaison VPC. Cela permet de AWS KMS créer des points de terminaison d'interface pour votre service de point de terminaison VPC. S'il ne s' AWS KMS agit pas d'un principal autorisé, les tentatives de création d'un magasin de clés externe échoueront, `XksProxyVpcEndpointServiceNotFoundException` sauf exception.

Suivez les instructions de la rubrique Manage permissions (Gérer les autorisations) du Guide AWS PrivateLink . Utilisez la valeur obligatoire suivante.

Champ	Valeur
ARN	`cks.kms.<region>.amazonaws.com` Par exemple, `cks.kms.us-east-1.amazonaws.com`

Suivant : Création d'un magasin de clés externe

Création d'un magasin de clés externe

Vous pouvez créer un ou plusieurs magasins de clés externes dans chaque Compte AWS région. Chaque magasin de clés externe doit être associé à un gestionnaire de clés externe et à un proxy de AWS magasin de clés externe (proxy XKS) qui assure la communication entre AWS KMS et votre gestionnaire de clés externe. Pour en savoir plus, consultez Choisissez une option de connectivité proxy pour un magasin de clés externe. Avant de commencer, vérifiez que vous avez besoin d'un magasin de clés externe. La plupart des clients peuvent utiliser des clés KMS soutenues par du matériel AWS KMS clé.

> **Tip**
>
> Certains gestionnaires de clés externes proposent une méthode plus simple pour créer un magasin de clés externe. Pour en savoir plus, veuillez consulter la documentation de votre gestionnaire de clés externe.

Avant de créer votre magasin de clés externe, vous devez réunir les conditions préalables. Au cours du processus de création, vous définissez les propriétés de votre magasin de clés externe. Plus important encore, vous indiquez si votre magasin de clés externe AWS KMS utilise un point de terminaison public ou un service de point de terminaison VPC pour se connecter à son proxy de magasin de clés externe. Vous spécifiez également les détails de la connexion, notamment le point de terminaison URI du proxy et le chemin au sein de ce point de terminaison du proxy où les demandes d'API sont AWS KMS envoyées au proxy.

Considérations

- KMS ne peut pas communiquer IPv6 avec les banques de clés externes.
- Si vous utilisez une connectivité de point de terminaison public, assurez-vous qu'il AWS KMS peut communiquer avec votre proxy via Internet à l'aide d'une connexion HTTPS. Cela implique de configurer le protocole TLS sur le proxy de stockage de clés externe et de s'assurer que tous les pare-feux situés entre le proxy AWS KMS et le proxy autorisent le IPv4 trafic à destination et en provenance du port 443 du proxy. Lors de la création d'un magasin de clés externe connecté à un point de terminaison public, AWS KMS teste la connexion en envoyant une demande d'état au proxy du magasin de clés externe. Ce test vérifie que le point de terminaison est accessible et que votre proxy de magasin de clés externe acceptera une requête signée avec vos informations d'identification pour l'authentification du proxy de magasin de clés externe. Si cette requête de test échoue, l'opération de création du magasin de clés externe échoue.
- Si vous utilisez la connectivité au service de point de terminaison d'un VPC, assurez-vous que l'équilibreur de charge réseau, le nom DNS privé et le service de point de terminaison d'un VPC sont correctement configurés et opérationnels. Si le proxy de banque de clés externe ne se trouve pas dans le VPC, vous devez vous assurer que le service de point de terminaison du VPC peut communiquer avec le proxy de banque de clés externe. (AWS KMS teste la connectivité du service de point de terminaison VPC lorsque vous connectez le magasin de clés externe à son proxy de magasin de clés externe.)
- AWS KMS enregistre les CloudWatch statistiques et les dimensions d'Amazon, en particulier pour les principaux magasins externes. Des graphiques de surveillance basés sur certaines de ces mesures apparaissent dans la AWS KMS console pour chaque magasin de clés externe. Nous vous recommandons vivement d'utiliser ces mesures pour créer des alarmes qui surveillent votre magasin de clés externe. Ces alertes vous préviennent des signes précoces de problèmes de performance et de fonctionnement avant qu'ils ne se produisent. Pour obtenir des instructions, veuillez consulter Surveillez les magasins de clés externes.

- Les magasins de clés externes sont soumis à des quotas de ressources. L'utilisation de clés KMS dans un magasin de clés externe est soumise à des quotas de requêtes. Passez en revue ces quotas avant de concevoir l'implémentation de votre magasin de clés externe.

> **Note**
>
> Vérifiez votre configuration pour détecter les dépendances circulaires susceptibles de l'empêcher de fonctionner.
> Par exemple, si vous créez votre proxy de stockage de clés externe à l'aide de AWS ressources, assurez-vous que le fonctionnement du proxy ne nécessite pas la disponibilité d'une clé KMS dans un magasin de clés externe accessible via ce proxy.

Tous les nouveaux magasins de clés externes sont créés dans un état déconnecté. Avant de créer des clés KMS dans votre magasin de clés externe, vous devez le connecter à son proxy de magasin de clés externe. Pour modifier les propriétés de votre magasin de clés externe, modifiez les paramètres de votre magasin de clés externe.

Rubriques

- Rassembler les conditions requises
- Création d'un nouveau magasin de clés externe

Rassembler les conditions requises

Avant de créer un magasin de clés externe, vous devez assembler les composants requis, notamment le gestionnaire de clés externe que vous utiliserez pour prendre en charge le magasin de clés externe et le proxy de magasin de clés externe qui traduit les AWS KMS demandes dans un format compréhensible par votre gestionnaire de clés externe.

Les composants suivants sont requis pour tous les magasins de clés externes. Outre ces composants, vous devez fournir les composants qui prennent en charge l'option de connectivité par proxy de magasin de clés externe que vous choisissez.

> **Tip**
>
> Votre gestionnaire de clés externe peut inclure certains de ces composants, ou ils peuvent être configurés pour vous. Pour en savoir plus, veuillez consulter la documentation de votre gestionnaire de clés externe.
> Si vous créez votre banque de clés externe dans la AWS KMS console, vous avez la possibilité de télécharger un fichier de configuration de proxy basé sur JSON qui spécifie le chemin de l'URI du proxy et les informations d'identification d'authentification du proxy. Certains proxys de magasin de clés externe génèrent ce fichier pour vous. Pour plus de détails, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre gestionnaire de clés externe.

Gestionnaire de clés externe

Chaque magasin de clés externe nécessite au moins une instance de gestionnaire de clés externe. Il peut s'agir d'un module de sécurité matérielle (HSM) physique ou virtuel ou d'un logiciel de gestion des clés.

Vous pouvez utiliser un seul gestionnaire de clés, mais nous recommandons au moins deux instances de gestionnaire de clés connexes qui partagent des clés cryptographiques pour des raisons de redondance. Le magasin de clés externe ne nécessite pas l'utilisation exclusive du gestionnaire de clés externe. Toutefois, le gestionnaire de clés externe doit être en mesure de gérer la fréquence prévue des demandes de chiffrement et de déchiffrement émanant des AWS services qui utilisent des clés KMS dans le magasin de clés externe afin de protéger vos ressources. Votre gestionnaire de clés externe doit être configuré pour traiter jusqu'à 1 800 requêtes par seconde et pour répondre dans le délai d'expiration de 250 millisecondes pour chaque requête. Nous vous recommandons de placer le gestionnaire de clés externe à proximité et de Région AWS manière à ce que le temps d'aller-retour (RTT) du réseau soit inférieur ou égal à 35 millisecondes.

Si le proxy de votre magasin de clés externe le permet, vous pouvez modifier le gestionnaire de clés externe que vous associez à votre proxy de magasin de clés externe, mais le nouveau gestionnaire de clés externe doit être une sauvegarde ou un instantané contenant les mêmes éléments de clé. Si la clé externe que vous associez à une clé KMS n'est plus disponible pour votre proxy de stockage de clés externe, vous AWS KMS ne pouvez pas déchiffrer le texte chiffré avec la clé KMS.

Le gestionnaire de clés externe doit être accessible au proxy de magasin de clés externe. Si la GetHealthStatusréponse du proxy indique que toutes les instances du gestionnaire de clés externe le

sont`Unavailable`, toutes les tentatives de création d'une banque de clés externe échouent avec un `XksProxyUriUnreachableException`.

Proxy de magasin de clés externe

Vous devez spécifier un proxy de magasin de clés externe (proxy XKS) conforme aux exigences de conception de spécification de l'API du proxy de magasin de clés externe AWS KMS (langue française non garantie). Vous pouvez développer ou acheter un proxy de stockage de clés externe, ou utiliser un proxy de stockage de clés externe fourni par ou intégré à votre gestionnaire de clés externe. AWS KMS recommande que votre proxy de stockage de clés externe soit configuré pour traiter jusqu'à 1 800 demandes par seconde et répondre dans le délai de 250 millisecondes pour chaque demande. Nous vous recommandons de placer le gestionnaire de clés externe à proximité et de Région AWS manière à ce que le temps d'aller-retour (RTT) du réseau soit inférieur ou égal à 35 millisecondes.

Vous pouvez utiliser un proxy de magasin de clés externe pour plusieurs magasins de clés externes, mais chaque magasin de clés externe doit disposer d'un point de terminaison et d'un chemin d'URI uniques au sein du proxy de magasin de clés externe pour ses requêtes.

Si vous utilisez la connectivité au service de point de terminaison d'un VPC, vous pouvez localiser le proxy de votre magasin de clés externe dans votre Amazon VPC, mais cela n'est pas obligatoire. Vous pouvez localiser votre proxy à l'extérieur AWS, par exemple dans votre centre de données privé, et utiliser le service de point de terminaison VPC uniquement pour communiquer avec le proxy.

Informations d'identification pour l'authentification du proxy

Pour créer un magasin de clés externe, vous devez spécifier vos informations d'identification pour l'authentification du proxy de magasin de clés externe (`XksProxyAuthenticationCredential`).

Vous devez établir un identifiant d'authentification (`XksProxyAuthenticationCredential`) pour votre proxy AWS KMS de magasin de clés externe. AWS KMS s'authentifie auprès de votre proxy en signant ses demandes à l'aide du processus Signature Version 4 (SigV4) avec les informations d'identification d'authentification du proxy de stockage de clés externe. Vous spécifiez les informations d'identification pour l'authentification lorsque vous créez votre magasin de clés externe et vous pouvez les modifier à tout moment. Si votre proxy effectue une rotation de vos informations d'identification, veillez à mettre à jour les valeurs d'informations d'identification de votre magasin de clés externe.

Les informations d'identification pour l'authentification du proxy comportent deux parties. Vous devez fournir les deux parties pour votre magasin de clés externe.

- ID de la clé d'accès : identifie la clé d'accès secrète. Vous pouvez fournir cet ID en texte brut.
- Clé d'accès secrète : partie secrète de l'identifiant. AWS KMS chiffre la clé d'accès secrète contenue dans les informations d'identification avant de les stocker.

Les informations d'identification SigV4 AWS KMS utilisées pour signer les demandes adressées au proxy de stockage de clés externe ne sont pas liées aux informations d'identification SigV4 associées aux AWS Identity and Access Management principaux de vos comptes. AWS Ne réutilisez aucune information d'identification IAM SigV4 pour votre proxy de magasin de clés externe.

Connectivité de proxy

Pour créer un magasin de clés externe, vous devez spécifier l'option de connectivité de proxy de votre magasin de clés externe (`XksProxyConnectivity`).

AWS KMS peut communiquer avec votre proxy de stockage de clés externe à l'aide d'un point de terminaison public ou d'un service de point de terminaison Amazon Virtual Private Cloud (Amazon VPC). Bien qu'un point de terminaison public soit plus simple à configurer et à gérer, il se peut qu'il ne réponde pas aux exigences de sécurité de chaque installation. Si vous choisissez l'option de connectivité au service de point de terminaison d'un Amazon VPC, vous devez créer et gérer les composants requis, notamment un Amazon VPC avec au moins deux sous-réseaux dans deux zones de disponibilité différentes, un service de point de terminaison d'un VPC avec un équilibreur de charge réseau et un groupe cible, ainsi qu'un nom DNS privé pour le service de point de terminaison d'un VPC.

Vous pouvez modifier l'option de connectivité de proxy pour votre magasin de clés externe. Toutefois, vous devez vous assurer de la disponibilité continue des éléments de clé associés aux clés KMS dans votre magasin de clés externe. Sinon, AWS KMS impossible de déchiffrer le texte chiffré avec ces clés KMS.

Pour savoir quelle option de connectivité de proxy convient le mieux à votre magasin de clés externe, veuillez consulter la rubrique Choisissez une option de connectivité proxy pour un magasin de clés externe. Pour obtenir de l'aide sur la création et la configuration de la connectivité au service de point de terminaison d'un VPC, veuillez consulter la rubrique Configurer la connectivité du service de point de terminaison VPC.

Point de terminaison d'URI de proxy

Pour créer un magasin de clés externe, vous devez spécifier le point de terminaison (`XksProxyUriEndpoint`) AWS KMS utilisé pour envoyer des demandes au proxy du magasin de clés externe.

Le protocole doit être HTTPS. AWS KMS communique IPv4 sur le port 443. Ne spécifiez pas le port dans la valeur de point de terminaison d'URI de proxy.

- Connectivité des points de terminaison publics : spécifiez le point de terminaison accessible au public pour votre proxy de magasin de clés externe. Ce point de terminaison doit être accessible avant de créer votre magasin de clés externe.
- Connectivité au service de point de terminaison d'un VPC : spécifiez `https://` suivi du nom DNS privé du service de point de terminaison d'un VPC.

Le certificat de serveur TLS configuré sur le proxy de magasin de clés externe doit correspondre au nom de domaine indiqué dans le point de terminaison de l'URI de proxy de magasin de clés externe et être émis par une autorité de certification prise en charge pour les magasins de clés externes. Pour obtenir une liste, veuillez consulter les Autorités de certification approuvées (langue française non garantie). Votre autorité de certification exigera une preuve de propriété du domaine avant de délivrer le certificat TLS.

Le nom commun (CN) du sujet sur le certificat TLS doit correspondre au nom DNS privé. Par exemple, si le nom DNS privé est `myproxy-private.xks.example.com`, le CN du certificat TLS doit être `myproxy-private.xks.example.com` ou `*.xks.example.com`.

Vous pouvez modifier le point de terminaison de l'URI de votre proxy, mais assurez-vous que le proxy de magasin de clés externe a accès aux éléments de clé associés aux clés KMS de votre magasin de clés externe. Sinon, AWS KMS impossible de déchiffrer le texte chiffré avec ces clés KMS.

Exigences relatives à l'unicité

- La combinaison du point de terminaison d'URI de proxy (`XksProxyUriEndpoint`) et de la valeur du chemin d'URI de proxy (`XksProxyUriPath`) doit être unique dans l' Compte AWS et la région.
- Les magasins de clés externes connectés à un point de terminaison public peuvent partager le même point de terminaison d'URI de proxy, à condition qu'ils aient des valeurs de chemin d'URI de proxy différentes.

- Un magasin de clés externe connecté à un point de terminaison public ne peut pas utiliser la même valeur de point de terminaison d'URI proxy qu'un magasin de clés externe doté d'une connectivité aux services de point de terminaison VPC Région AWS, même si les magasins de clés se trouvent dans des emplacements différents. Comptes AWS
- Chaque magasin de clés externe connecté à un point de terminaison d'un VPC doit avoir son propre nom DNS privé. Le point de terminaison URI du proxy (nom DNS privé) doit être unique dans la région Compte AWS et.

Chemin d'URI de proxy

Pour créer un magasin de clés externe, vous devez spécifier le chemin de base dans votre proxy de magasin de clés externe vers le proxy requis APIs. La valeur doit commencer par `/` et se terminer par/kms/xks/v1, qui `v1` représente la version de l' AWS KMS API pour le proxy de stockage de clés externe. Ce chemin peut inclure un préfixe facultatif entre les éléments requis tels que `/example-prefix/kms/xks/v1`. Pour trouver cette valeur, veuillez consulter la documentation de votre proxy de magasin de clés externe.

AWS KMS envoie des demandes de proxy à l'adresse spécifiée par la concaténation du point de terminaison de l'URI du proxy et du chemin de l'URI du proxy. Par exemple, si le point de terminaison de l'URI du proxy est `https://myproxy.xks.example.com` et que le chemin de l'URI du proxy est`/kms/xks/v1`, AWS KMS envoie ses demandes d'API proxy à`https://myproxy.xks.example.com/kms/xks/v1`.

Vous pouvez modifier le chemin d'URI de votre proxy, mais assurez-vous que le proxy de magasin de clés externe a accès aux éléments de clé associés aux clés KMS de votre magasin de clés externe. Sinon, AWS KMS impossible de déchiffrer le texte chiffré avec ces clés KMS.

Exigences relatives à l'unicité

- La combinaison du point de terminaison d'URI de proxy (`XksProxyUriEndpoint`) et de la valeur du chemin d'URI de proxy (`XksProxyUriPath`) doit être unique dans l' Compte AWS et la région.

Service de point de terminaison d’un VPC

Spécifie le nom du service de point de terminaison d'un Amazon VPC utilisé pour communiquer avec le proxy de votre magasin de clés externe. Ce composant n'est requis que pour les magasins de clés externes qui utilisent la connectivité au service de point de terminaison d'un VPC. Pour obtenir de l'aide sur la configuration de votre service de point de terminaison d'un VPC pour un magasin de clés

externe, veuillez consulter la rubrique Configurer la connectivité du service de point de terminaison VPC.

Le service de point de terminaison d'un VPC doit posséder les propriétés suivantes :

- Le service de point de terminaison VPC doit se trouver dans la même Compte AWS région que le magasin de clés externe.
- Il doit comporter un équilibreur de charge réseau (NLB) connecté à au moins deux sous-réseaux, dans deux zones de disponibilités distinctes.
- La liste des principaux autorisés pour le service de point de terminaison VPC doit inclure AWS KMS le principal de service pour la région `cks.kms.<region>.amazonaws.com` :, tel que. `cks.kms.us-east-1.amazonaws.com`
- L'acceptation des requêtes de connexion ne doit pas être requise.
- Il doit avoir un nom DNS privé dans un domaine public de niveau supérieur. Par exemple, vous pouvez avoir un nom DNS privé myproxy-private.xks.example.com dans le domaine public `xks.example.com`.

  Le nom DNS privé d'un magasin de clés externe doté d'une connectivité au service de point de terminaison d'un VPC doit être unique dans sa Région AWS.
- L'état de vérification du domaine du nom DNS privé doit être `verified`.
- Le certificat de serveur TLS configuré sur le proxy de magasin de clés externe doit spécifier le nom d'hôte DNS privé auquel le point de terminaison est accessible.

Exigences relatives à l'unicité

- Les magasins de clés externes connectés à des points de terminaison d'un VPC peuvent partager un `Amazon VPC`, mais chaque magasin de clés externe doit disposer de son propre service de point de terminaison d'un VPC et d'un nom DNS privé.

Fichier de configuration du proxy

Un fichier de configuration de proxy est un fichier JSON facultatif qui contient des valeurs pour le chemin d'URI de proxy et les propriétés d'informations d'identification pour l'authentification du proxy de votre magasin de clés externe. Lorsque vous créez ou modifiez un magasin de clés externe dans la console AWS KMS , vous pouvez télécharger un fichier de configuration de proxy pour fournir des valeurs de configuration pour votre magasin de clés externe. L'utilisation de ce fichier

évite les erreurs de saisie et de collage et garantit que les valeurs de votre magasin de clés externe correspondent à celles de votre proxy de magasin de clés externe.

Les fichiers de configuration du proxy sont générés par le proxy de magasin de clés externe. Pour savoir si votre proxy de magasin de clés externe propose un fichier de configuration de proxy, veuillez consulter la documentation relative à votre proxy de magasin de clés externe.

Voici un exemple de fichier de configuration de proxy correctement formaté avec des valeurs fictives.

```
{
  "XksProxyUriPath": "/example-prefix/kms/xks/v1",
  "XksProxyAuthenticationCredential": {
    "AccessKeyId": "ABCDE12345670EXAMPLE",
    "RawSecretAccessKey": "0000EXAMPLEFA5FT0mCc3DrGUe2sti527BitkQ0Zr9MO9+vE="
  }
}
```

Vous pouvez télécharger un fichier de configuration de proxy uniquement lors de la création ou de la modification d'un magasin de clés externe dans la AWS KMS console. Vous ne pouvez pas l'utiliser avec les UpdateCustomKeyStoreopérations CreateCustomKeyStoreor, mais vous pouvez utiliser les valeurs du fichier de configuration du proxy pour vous assurer que les valeurs de vos paramètres sont correctes.

Création d'un nouveau magasin de clés externe

Une fois que vous avez réuni les prérequis nécessaires, vous pouvez créer un nouveau magasin de clés externe dans la AWS KMS console ou en utilisant l'CreateCustomKeyStoreopération.

Utilisation de la AWS KMS console

Avant de créer un magasin de clés externe, choisissez votre type de connectivité proxy et assurez-vous d'avoir créé et configuré tous les composants requis. Si vous avez besoin d'aide pour trouver l'une des valeurs requises, consultez la documentation de votre proxy de magasin de clés externe ou de votre logiciel de gestion des clés.

> **Note**
>
> Lorsque vous créez un magasin de clés externe dans le AWS Management Console, vous pouvez télécharger un fichier de configuration de proxy basé sur JSON avec des valeurs pour

le chemin de l'URI du proxy et les informations d'identification d'authentification du proxy. Certains proxys génèrent ce fichier pour vous. Il n'est pas obligatoire.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Choisissez Create external key store (Créer un magasin de clés externe).
5. Saisissez un nom convivial pour le magasin de clés externe. Le nom doit être unique parmi tous les magasins de clés externes de votre compte.

   > **Important**
   >
   > N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

6. Choisissez votre type de connectivité de proxy.

   Votre choix de connectivité de proxy détermine les composants requis pour votre proxy de magasin de clés externe. Pour obtenir de l'aide pour faire ce choix, veuillez consulter la rubrique Choisissez une option de connectivité proxy pour un magasin de clés externe.
7. Choisissez ou saisissez le nom du service de point de terminaison d'un VPC pour ce magasin de clés externe. Cette étape s'affiche uniquement lorsque le type de connectivité du proxy de votre magasin de clés externe est VPC endpoint service (Service de point de terminaison d'un VPC).

   Le service de point de terminaison VPC et son service VPCs doivent répondre aux exigences d'un magasin de clés externe. Pour en savoir plus, consultez the section called “Rassembler les conditions requises”.
8. Saisissez votre point de terminaison d'URI de proxy. Le protocole doit être HTTPS. AWS KMS communique IPv4 sur le port 443. Ne spécifiez pas le port dans la valeur de point de terminaison d'URI de proxy.

   S'il AWS KMS reconnaît le service de point de terminaison VPC que vous avez spécifié à l'étape précédente, il complète ce champ pour vous.

Pour la connectivité au point de terminaison public, saisissez un URI de point de terminaison accessible au public. Pour la connectivité au point de terminaison d'un VPC, saisissez `https://` suivi du nom DNS privé du service de point de terminaison d'un VPC.

9. Pour saisir les valeurs du préfixe du chemin d'URI de proxy et des informations d'identification pour l'authentification du proxy, chargez un fichier de configuration de proxy ou saisissez les valeurs manuellement.

   - Si vous disposez d'un fichier de configuration de proxy facultatif contenant des valeurs pour le chemin d'URI de votre proxy et les informations d'identification pour l'authentification du proxy, choisissez Upload configuration file (Charger le fichier de configuration). Suivez les instructions pour charger le fichier.

     Lorsque le fichier est chargé, la console affiche les valeurs du fichier dans des champs modifiables. Vous pouvez changer les valeurs maintenant ou modifier ces valeurs après la création du magasin de clés externe.

     Pour afficher la valeur de la clé d'accès secrète, choisissez Show secret access key (Afficher la clé d'accès secrète).

   - Si vous ne disposez pas d'un fichier de configuration du proxy, vous pouvez saisir manuellement le chemin d'URI de proxy et les valeurs d'informations d'identification pour l'authentification du proxy.

     a. Si vous n'avez pas de fichier de configuration de proxy, vous pouvez saisir manuellement l'URI de votre proxy. La console fournit la valeur/kms/xks/v1 requise.

        Si votre chemin d'URI de proxy inclut un préfixe facultatif, tel que l'`example-prefix` dans `/example-prefix/kms/xks/v1`, saisissez le préfixe dans le champ Proxy URI path prefix (Préfixe du chemin d'URI de proxy). Sinon, laissez le champ vide.

     b. Si vous ne disposez pas d'un fichier de configuration du proxy, vous pouvez saisir vos informations d'identification pour l'authentification du proxy manuellement. L'ID de clé d'accès et la clé d'accès secrète sont tous deux requis.

        - Dans Proxy credential: Access key ID (Informations d'identification du proxy : identifiant de la clé d'accès), saisissez l'ID de clé d'accès des informations d'identification pour l'authentification du proxy. L'ID de clé d'accès identifie la clé d'accès secrète.

- Dans Proxy credential: Secret access key (Informations d'identification du proxy : clé d'accès secrète), saisissez la clé d'accès secrète des informations d'identification pour l'authentification du proxy.

  Pour afficher la valeur de la clé d'accès secrète, choisissez Show secret access key (Afficher la clé d'accès secrète).

  Cette procédure ne définit ni ne modifie les informations d'identification pour l'authentification que vous avez établies sur votre proxy de magasin de clés externe. Elle associe simplement ces valeurs à votre magasin de clés externe. Pour plus d'informations sur la définition, la modification et la rotation de vos informations d'identification pour l'authentification du proxy, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre logiciel de gestion des clés.

  Si vos informations d'identification pour l'authentification du proxy changent, modifiez le paramètre des informations d'identification de votre magasin de clés externe.

10. Choisissez Create external key store (Créer un magasin de clés externe).

Lorsque la procédure se termine avec succès, le nouveau magasin de clés externe s'affiche dans la liste des magasins de clés externes du compte et de la région. S'il ne réussit pas, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez CreateKey erreurs pour la clé externe.

Suivant : les nouveaux magasins de clés externes ne sont pas automatiquement connectés. Avant de pouvoir créer AWS KMS keys dans votre magasin de clés externe, vous devez connecter le magasin de clés externe à son proxy de magasin de clés externe.

Utilisation de l' AWS KMS API

Vous pouvez utiliser cette CreateCustomKeyStoreopération pour créer un nouveau magasin de clés externe. Pour obtenir de l'aide pour trouver les valeurs des paramètres requis, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre logiciel de gestion des clés.

> **Tip**
>
> Vous ne pouvez pas charger de fichier de configuration de proxy lors de l'utilisation de l'opération `CreateCustomKeyStore`. Vous pouvez toutefois utiliser les valeurs du fichier de configuration de proxy pour vous assurer que les valeurs de vos paramètres sont correctes.

Pour créer un magasin de clés externe, l'opération `CreateCustomKeyStore` nécessite les valeurs de paramètres suivantes.

- `CustomKeyStoreName` : un nom convivial pour le magasin de clés externe qui est unique dans le compte.

  > **Important**
  >
  > N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.

- `CustomKeyStoreType` : spécifiez `EXTERNAL_KEY_STORE`.
- `XksProxyConnectivity` : spécifiez `PUBLIC_ENDPOINT` ou `VPC_ENDPOINT_SERVICE`.
- `XksProxyAuthenticationCredential` : spécifiez à la fois l'ID de clé d'accès et la clé d'accès secrète.
- `XksProxyUriEndpoint`— Le point de terminaison AWS KMS utilisé pour communiquer avec votre proxy de stockage de clés externe.
- `XksProxyUriPath`— Le chemin d'accès au proxy au sein du proxy APIs.
- `XksProxyVpcEndpointServiceName` : obligatoire uniquement lorsque la valeur de votre `XksProxyConnectivity` est `VPC_ENDPOINT_SERVICE`.

> **Note**
>
> Si vous utilisez AWS CLI la version 1.0, exécutez la commande suivante avant de spécifier un paramètre avec une valeur HTTP ou HTTPS, tel que le `XksProxyUriEndpoint` paramètre.
>
> ```
> aws configure set cli_follow_urlparam false
> ```

Dans le cas contraire, la AWS CLI version 1.0 remplace la valeur du paramètre par le contenu trouvé à cette adresse URI, ce qui provoque l'erreur suivante :

```
Error parsing parameter '--xks-proxy-uri-endpoint': Unable to retrieve
https:// : received non 200 status code of 404
```

Les exemples suivants utilisent des valeurs fictives. Avant d'exécuter la commande, remplacez-les par des valeurs valides pour votre magasin de clés externe.

Créez un magasin de clés externe avec une connectivité au point de terminaison public.

```
$ aws kms create-custom-key-store
        --custom-key-store-name ExampleExternalKeyStorePublic \
        --custom-key-store-type EXTERNAL_KEY_STORE \
        --xks-proxy-connectivity PUBLIC_ENDPOINT \
        --xks-proxy-uri-endpoint https://myproxy.xks.example.com \
        --xks-proxy-uri-path /kms/xks/v1 \
        --xks-proxy-authentication-credential
 AccessKeyId=<value>,RawSecretAccessKey=<value>
```

Créez un magasin de clés externe avec une connectivité au service de point de terminaison d'un VPC.

```
$ aws kms create-custom-key-store
        --custom-key-store-name ExampleExternalKeyStoreVPC \
        --custom-key-store-type EXTERNAL_KEY_STORE \
        --xks-proxy-connectivity VPC_ENDPOINT_SERVICE \
        --xks-proxy-vpc-endpoint-service-name com.amazonaws.vpce.us-east-1.vpce-svc-
example \
        --xks-proxy-uri-endpoint https://myproxy-private.xks.example.com \
        --xks-proxy-uri-path /kms/xks/v1 \
        --xks-proxy-authentication-credential
 AccessKeyId=<value>,RawSecretAccessKey=<value>
```

Lorsque l'opération est réussie, `CreateCustomKeyStore` renvoie l'ID du magasin de clés personnalisé, comme illustré dans l'exemple de réponse suivant.

```
{
```

```
    "CustomKeyStoreId": cks-1234567890abcdef0
}
```

Si l'opération échoue, corrigez l'erreur indiquée par l'exception, puis réessayez. Pour obtenir de l'aide supplémentaire, consultez Résoudre les problèmes liés aux magasins de clés externes.

Suivant : pour utiliser le magasin de clés externe, connectez-le à son proxy de magasin de clés externe.

Modifier les propriétés du magasin de clés externe

Vous pouvez modifier les propriétés sélectionnées d'un magasin de clés externe existant.

Vous pouvez modifier certaines propriétés lorsque le magasin de clés externe est connecté ou déconnecté. Pour les autres propriétés, vous devez d'abord déconnecter votre magasin de clés externe de son proxy de magasin de clés externe. L'état de connexion du magasin de clés externe doit être `DISCONNECTED`. Lorsque votre magasin de clés externe est déconnecté, vous pouvez gérer le magasin de clés et ses clés KMS, mais vous ne pouvez pas créer ou utiliser des clés KMS dans le magasin de clés externe. Pour connaître l'état de connexion de votre magasin de clés externe, utilisez l'DescribeCustomKeyStoresopération ou consultez la section Configuration générale sur la page détaillée du magasin de clés externe.

Avant de mettre à jour les propriétés de votre magasin de clés externe, AWS KMS envoie une GetHealthStatusdemande au proxy du magasin de clés externe en utilisant les nouvelles valeurs. Si la requête aboutit, cela indique que vous pouvez vous connecter et vous authentifier à un proxy de magasin de clés externe avec les valeurs de propriété mises à jour. Si la requête échoue, l'opération de modification échoue avec une exception qui identifie l'erreur.

Lorsque l'opération de modification est terminée, les valeurs de propriété mises à jour pour votre magasin de clés externe apparaissent dans la AWS KMS console et dans la `DescribeCustomKeyStores` réponse. Toutefois, il peut s'écouler jusqu'à cinq minutes avant que les modifications ne soient pleinement effectives.

Si vous modifiez votre banque de clés externe dans la AWS KMS console, vous avez la possibilité de télécharger un fichier de configuration de proxy basé sur JSON qui spécifie le chemin de l'URI du proxy et les informations d'identification d'authentification du proxy. Certains proxys de magasin de clés externe génèrent ce fichier pour vous. Pour plus de détails, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre gestionnaire de clés externe.

> **Warning**
>
> Les valeurs de propriété mises à jour doivent connecter votre magasin de clés externe à un proxy pour le même gestionnaire de clés externe que celui utilisé dans les valeurs précédentes, ou pour une sauvegarde ou un instantané du gestionnaire de clés externe avec les mêmes clés cryptographiques. Si votre magasin de clés externe perd définitivement l'accès aux clés externes associées à ses clés KMS, le texte chiffré qui a été chiffré au moyen de ces clés externes est irrécupérable. En particulier, la modification de la connectivité proxy d'un magasin de clés externe peut AWS KMS empêcher l'accès à vos clés externes.

> **Tip**
>
> Certains gestionnaires de clés externes proposent une méthode plus simple pour modifier les propriétés du magasin de clés externe. Pour en savoir plus, veuillez consulter la documentation de votre gestionnaire de clés externe.

Vous pouvez modifier les propriétés suivantes d'un magasin de clés externe.

Propriétés du magasin de clés externe modifiables	Tout état de connexion	Exiger l'état Déconnecté
Nom du magasin de clés personnalisé Un nom convivial requis pour un magasin de clés personnalisé. **Important** N'incluez pas d'informations confidentielles ou sensibles dans ce champ. Ce champ peut être affiché en texte brut dans les CloudTrail journaux et autres sorties.	✓	
Identifiant d'authentification du proxy () XksProxyAuthenticationCredential	✓	

Propriétés du magasin de clés externe modifiables	Tout état de connexion	Exiger l'état Déconnecté
(Vous devez spécifier à la fois l'ID de clé d'accès et la clé d'accès secrète, même si vous ne modifiez qu'un seul élément.)		
Chemin de l'URI du proxy (XksProxyUriPath)	✓	
Connectivité proxy (XksProxyConnectivity) (Vous devez également mettre à jour le point de terminaison d'URI de proxy. Si vous passez à la connectivité au service de point de terminaison d'un VPC, vous devez spécifier un nom de service de point de terminaison d'un VPC proxy.)		✓
Point de terminaison URI du proxy (XksProxy UriEndpoint) Si vous modifiez l'URI du point de terminaison du proxy, vous devrez peut-être aussi modifier le certificat TLS associé.		✓
Nom du service de point de terminaison VPC proxy () XksProxyVpcEndpointServiceName (Ce champ est obligatoire pour la connectivité au service de point de terminaison d'un VPC)		✓

Modifier les propriétés de votre magasin de clés externe

Vous pouvez modifier les propriétés de votre magasin de clés externe dans la AWS KMS console ou en utilisant cette UpdateCustomKeyStoreopération.

Utilisation de la AWS KMS console

Lorsque vous modifiez un magasin de clés, vous pouvez changer n'importe laquelle des valeurs modifiables. Certaines modifications nécessitent que le magasin de clés externe soit déconnecté de son proxy de magasin de clés externe.

Si vous modifiez le chemin d'URI de proxy ou les informations d'identification pour l'authentification du proxy, vous pouvez saisir les nouvelles valeurs ou charger un fichier de configuration de proxy de magasin de clés externe qui contient les nouvelles valeurs.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Choisissez la ligne du magasin de clés externe que vous souhaitez modifier.
5. Si nécessaire, déconnectez le magasin de clés externe de son proxy de magasin de clés externe. Dans le menu Key store actions (Actions de magasin de clés), choisissez Disconnect (Déconnecter).
6. À partir du menu Key store actions (Actions de magasin de clés), choisissez Edit (Modifier).
7. Modifiez une ou plusieurs propriétés modifiables du magasin de clés externe. Vous pouvez également charger un fichier de configuration de proxy de magasin de clés externe contenant des valeurs pour le chemin d'URI de proxy et les informations d'identification pour l'authentification du proxy. Vous pouvez utiliser un fichier de configuration de proxy même si certaines valeurs spécifiées dans le fichier n'ont pas changé.
8. Choisissez Update external key store (Mettre à jour le magasin de clés externe).
9. Passez en revue l'avertissement et, si vous décidez de continuer, confirmez-le, puis choisissez Update external key store (Mettre à jour le magasin de clés externe).

   Lorsque la procédure se déroule avec succès, un message décrit les propriétés que vous avez modifiées. Si elle ne réussit pas, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre.
10. Si nécessaire, reconnectez le magasin de clés externe. Dans le menu Key store actions (Actions de magasin de clés), choisissez Connect (Connecter).

Vous pouvez laisser le magasin de clés externe déconnecté. Mais, tant qu'il est déconnecté, vous ne pouvez pas créer de clés KMS dans le magasin de clés externe ou utiliser les clés KMS du magasin de clés externe pour les opérations cryptographiques.

Utilisation de l' AWS KMS API

Pour modifier les propriétés d'un magasin de clés externe, utilisez l'UpdateCustomKeyStoreopération. Vous pouvez modifier plusieurs propriétés d'un magasin de clés externe dans la même opération. Si l'opération aboutit, AWS KMS renvoie une réponse HTTP 200 et un objet JSON sans propriétés.

Utilisez le paramètre `CustomKeyStoreId` pour identifier le magasin de clés externe. Utilisez les autres paramètres pour modifier les propriétés. Vous ne pouvez pas utiliser de fichier de configuration de proxy pour l'opération `UpdateCustomKeyStore`. Le fichier de configuration du proxy n'est pris en charge que par la AWS KMS console. Vous pouvez toutefois utiliser le fichier de configuration du proxy pour vous aider à déterminer les valeurs de paramètres correctes pour le proxy de votre magasin de clés externe.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Avant de commencer, si nécessaire, déconnectez le magasin de clés externe de son proxy de magasin de clés externe. Après la mise à jour, vous pouvez, si nécessaire, reconnecter le magasin de clés externe à son proxy de magasin de clés externe. Vous pouvez laisser le magasin de clés externe à l'état déconnecté, mais vous devez le reconnecter avant de pouvoir créer des clés KMS dans le magasin de clés ou d'utiliser les clés KMS existantes du magasin de clés pour les opérations cryptographiques.

> **Note**
>
> Si vous utilisez AWS CLI la version 1.0, exécutez la commande suivante avant de spécifier un paramètre avec une valeur HTTP ou HTTPS, tel que le `XksProxyUriEndpoint` paramètre.
>
> ```
> aws configure set cli_follow_urlparam false
> ```
>
> Dans le cas contraire, la AWS CLI version 1.0 remplace la valeur du paramètre par le contenu trouvé à cette adresse URI, ce qui provoque l'erreur suivante :

```
Error parsing parameter '--xks-proxy-uri-endpoint': Unable to retrieve
https:// : received non 200 status code of 404
```

Modifier le nom du magasin de clés externe

Le premier exemple utilise l'UpdateCustomKeyStoreopération pour changer le nom convivial du magasin de clés externe en`XksKeyStore`. La commande utilise le paramètre `CustomKeyStoreId` pour identifier le magasin de clés personnalisé et le paramètre `CustomKeyStoreName` pour spécifier le nouveau nom du magasin de clés personnalisé. Remplacez toutes les valeurs d'exemple par des valeurs réelles pour votre magasin de clés externe.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --new-
custom-key-store-name XksKeyStore
```

Modifier les informations d'identification pour l'authentification du proxy

L'exemple suivant met à jour les informations d'identification d'authentification du proxy AWS KMS utilisées pour s'authentifier auprès du proxy de stockage de clés externe. Vous pouvez utiliser une commande comme celle-ci pour effectuer une rotation des informations d'identification si elles ont subi une rotation sur votre proxy.

Mettez d'abord à jour les informations d'identification sur le proxy de votre magasin de clés externe. Utilisez ensuite cette fonctionnalité pour signaler la modification à AWS KMS. (Votre proxy prendra brièvement en charge l'ancienne et la nouvelle identification afin que vous ayez le temps de mettre à jour vos informations d'identification.) AWS KMS

Vous devez toujours spécifier à la fois l'ID de la clé d'accès et la clé d'accès secrète dans les informations d'identification, même si une seule valeur est modifiée.

Les deux premières commandes définissent des variables pour contenir les valeurs des informations d'identification. Les opérations `UpdateCustomKeyStore` utilisent le paramètre `CustomKeyStoreId` pour identifier le magasin de clés externe. Il utilise le paramètre `XksProxyAuthenticationCredential` avec ses champs `AccessKeyId` et `RawSecretAccessKey` pour spécifier les nouvelles informations d'identification. Remplacez toutes les valeurs d'exemple par des valeurs réelles pour votre magasin de clés externe.

```
$ accessKeyID=access key id
```

```
$ secretAccessKey=secret access key

$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
        --xks-proxy-authentication-credential \
            AccessKeyId=$accessKeyId,RawSecretAccessKey=$secretAccessKey
```

Modifier le chemin d'URI de proxy

L'exemple suivant met à jour le chemin d'URI de proxy (`XksProxyUriPath`). La combinaison du point de terminaison d'URI de proxy et du chemin d'URI de proxy doit être unique dans l' Compte AWS et la région. Remplacez toutes les valeurs d'exemple par des valeurs réelles pour votre magasin de clés externe.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
            --xks-proxy-uri-path /kms/xks/v1
```

Modifier la connectivité au service de point de terminaison d'un VPC

L'exemple suivant utilise l'UpdateCustomKeyStoreopération pour modifier le type de connectivité du proxy du magasin de clés externe en`VPC_ENDPOINT_SERVICE`. Pour effectuer cette modification, vous devez spécifier les valeurs requises pour la connectivité au service de point de terminaison d'un VPC, notamment le nom du service de point de terminaison d'un VPC (`XksProxyVpcEndpointServiceName`) et une valeur de point de terminaison d'URI de proxy (`XksProxyUriEndpoint`) qui inclut le nom DNS privé du service de point de terminaison d'un VPC. Remplacez toutes les valeurs d'exemple par des valeurs réelles pour votre magasin de clés externe.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
            --xks-proxy-connectivity "VPC_ENDPOINT_SERVICE" \
            --xks-proxy-uri-endpoint https://myproxy-private.xks.example.com \
            --xks-proxy-vpc-endpoint-service-name com.amazonaws.vpce.us-east-1.vpce-
svc-example
```

Passer à une connectivité au point de terminaison public

L'exemple suivant remplace le type de connectivité du proxy de magasin de clés externe par `PUBLIC_ENDPOINT`. Lorsque vous effectuez cette modification, vous devez mettre à jour la valeur du point de terminaison de l'URI de proxy (`XksProxyUriEndpoint`). Remplacez toutes les valeurs d'exemple par des valeurs réelles pour votre magasin de clés externe.

> **Note**
>
> La connectivité au point de terminaison d'un VPC offre une plus grande sécurité que la connectivité au point de terminaison public. Avant de passer à la connectivité au point de terminaison public, envisagez d'autres options, notamment la localisation de votre proxy de magasin de clés externe sur site et l'utilisation du VPC uniquement pour la communication.

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
            --xks-proxy-connectivity "PUBLIC_ENDPOINT" \
            --xks-proxy-uri-endpoint https://myproxy.xks.example.com
```

Afficher les magasins de clés externes

Vous pouvez consulter les banques de clés externes de chaque compte et de chaque région à l'aide de la AWS KMS console ou à l'aide de l'DescribeCustomKeyStoresopération.

Lorsque vous consultez un magasin de clés externe, vous pouvez voir les éléments suivants :

- Des informations de base sur le magasin de clés, notamment son nom convivial, son ID, son type de magasin de clés et sa date de création.
- Des informations de configuration pour le proxy de magasin de clés externe, notamment le type de connectivité, le point de terminaison et le chemin d'URI de proxy, ainsi que l'ID de clé d'accès de vos informations d'identification pour l'authentification du proxy actuelles.
- Si le proxy de magasin de clés externe utilise la connectivité au service de point de terminaison d'un VPC, la console affiche le nom du service de point de terminaison d'un VPC.
- L'état de la connexion actuel.

> **Note**
>
> La valeur d'état de connexion Disconnected (Déconnectée) indique que le magasin de clés externe n'a jamais été connecté ou qu'il a été intentionnellement déconnecté de son proxy de magasin de clés externe. Cependant, si vos tentatives d'utiliser une clé KMS dans un magasin de clés externe connecté échouent, cela peut signifier la présence d'un problème avec le magasin de clés externe ou son proxy. Pour obtenir de l'aide, veuillez consulter Erreurs de connexion au magasin de clés externe.

- Une section de surveillance contenant des graphiques des CloudWatch statistiques Amazon conçues pour vous aider à détecter et à résoudre les problèmes liés à votre magasin de clés externe. Pour obtenir de l'aide pour interpréter les graphiques, les utiliser dans le cadre de votre planification et de votre résolution des problèmes, et pour créer des CloudWatch alarmes en fonction des indicateurs présentés dans les graphiques, consultezSurveillez les magasins de clés externes.

Propriétés du magasin de clés externe

Les propriétés suivantes d'un magasin de clés externe sont visibles dans la AWS KMS console et dans la DescribeCustomKeyStoresréponse.

Propriétés du magasin de clés personnalisé

Les valeurs suivantes apparaissent dans la section Configuration générale de la page détaillée de chaque magasin de clés personnalisé. Ces propriétés s'appliquent à tous les magasins de clés personnalisés, y compris les magasins de AWS CloudHSM clés et les magasins de clés externes.

ID du magasin de clés personnalisé

Un identifiant unique AWS KMS attribué au magasin de clés personnalisé.

Nom du magasin de clés personnalisé

Un nom convivial que vous attribuez au magasin de clés personnalisé lorsque vous le créez. Vous pouvez modifier cette valeur à tout moment.

Type de magasin de clés personnalisé

Le type de magasin de clés personnalisé. Les valeurs valides sont AWS CloudHSM (`AWS_CLOUDHSM`) ou Magasin de clés externe (`EXTERNAL_KEY_STORE`). Vous ne pouvez pas modifier le type après avoir créé le magasin de clés personnalisé.

Date de création

La date à laquelle le magasin de clés personnalisé a été créé. Cette date est affichée en heure locale pour l' Région AWS.

État de connexion

Indique si le magasin de clés personnalisé est connecté au magasin de clés de sauvegarde. L'état de connexion est `DISCONNECTED` uniquement si le magasin de clés personnalisé n'a jamais été

connecté à son magasin de clés de sauvegarde, ou s'il a été déconnecté intentionnellement. Pour plus de détails, consultez the section called “État de connexion”.

Propriétés de configuration du magasin de clés externe

Les valeurs suivantes apparaissent dans la section Configuration du proxy du magasin de clés externe de la page détaillée de chaque magasin de clés externe et dans l'`XksProxyConfiguration`élément de DescribeCustomKeyStoresréponse. Pour obtenir une description détaillée de chaque champ, y compris les exigences d'unicité et de l'aide pour déterminer la valeur correcte de chaque champ, veuillez consulter the section called “Rassembler les conditions requises” dans la rubrique Créer un magasin de clés externe.

Connectivité de proxy

Indique si le magasin de clés externe utilise une connectivité au point de terminaison public ou une connectivité au service de point de terminaison d'un VPC.

Point de terminaison d'URI de proxy

Le point de terminaison AWS KMS utilisé pour se connecter à votre proxy de stockage de clés externe.

Chemin d'URI de proxy

Le chemin depuis le point de terminaison de l'URI du proxy où AWS KMS envoie les demandes d'API du proxy.

Informations d'identification du proxy : ID de la clé d'accès

Fait partie des informations d'identification pour l'authentification du proxy que vous définissez sur votre proxy de magasin de clés externe. L'ID de clé d'accès identifie la clé d'accès secrète dans les informations d'identification.

AWS KMS utilise le processus de signature SigV4 et les informations d'identification d'authentification du proxy pour signer ses demandes à votre proxy de stockage de clés externe. Les informations d'identification contenues dans la signature permettent au proxy de stockage de clés externe d'authentifier les demandes en votre nom auprès de. AWS KMS

Nom du service de point de terminaison d'un VPC

Nom du service de point de terminaison d'un Amazon VPC prenant en charge votre magasin de clés externe. Cette valeur n'apparaît que lorsque le magasin de clés externe utilise la connectivité

au service de point de terminaison d'un VPC. Vous pouvez localiser votre proxy de magasin de clés externe dans le VPC ou utiliser le service de point de terminaison d'un VPC pour communiquer en toute sécurité avec votre proxy de magasin de clés externe.

Afficher les propriétés de votre magasin de clés externe

Vous pouvez consulter votre magasin de clés externe et ses propriétés associées dans la AWS KMS console ou en utilisant l'DescribeCustomKeyStoresopération.

Utilisation de la AWS KMS console

Pour consulter les magasins de clés externes d'un compte et d'une région donnés, utilisez la procédure suivante.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Pour consulter des informations détaillées sur un magasin de clés externe, sélectionnez le nom du magasin de clés.

Utilisation de l' AWS KMS API

Pour afficher vos stockages de clés externes, utilisez l'DescribeCustomKeyStoresopération. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Toutefois, vous pouvez utiliser le paramètre `CustomKeyStoreName` ou `CustomKeyStoreId` (mais pas les deux) pour limiter la sortie à un magasin de clés personnalisé en particulier.

Pour les magasins de clés personnalisées, la sortie contient l'ID, le nom et le type du magasin de clés personnalisé, ainsi que l'état de connexion du magasin de clés. Si l'état de connexion est `FAILED`, la sortie contient également un `ConnectionErrorCode` qui décrit la raison de l'erreur. Pour obtenir de l'aide pour interpréter le `ConnectionErrorCode` pour un magasin de clés externe, veuillez consulter la rubrique Codes d'erreur de connexion pour les magasins de clés externes.

Pour les magasins de clés externes, la sortie contient également l'élément `XksProxyConfiguration`. Cet élément inclut le type de connectivité, le point de terminaison

d'URI de proxy, le chemin d'URI de proxy et l'ID de clé d'accès des informations d'identification pour l'authentification du proxy.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Par exemple, la commande suivante renvoie tous les magasins de clés personnalisées du compte et de la région. Vous pouvez utiliser les paramètres `Marker` et `Limit` pour parcourir les magasins de clés personnalisés de la sortie.

```
$ aws kms describe-custom-key-stores
```

La commande suivante utilise le paramètre `CustomKeyStoreName` pour obtenir uniquement l'exemple de magasin de clés externe avec le nom convivial `ExampleXksPublic`. Cet exemple de magasin de clés utilise la connectivité au point de terminaison public. Il est connecté à son proxy de magasin de clés externe.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksPublic
{
    "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleXksPublic",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-14T20:17:36.419000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE12345670EXAMPLE",
        "Connectivity": "PUBLIC_ENDPOINT",
        "UriEndpoint": "https://xks.example.com:6443",
        "UriPath": "/example/prefix/kms/xks/v1"
      }
    }
  ]
}
```

La commande suivante permet d'obtenir un exemple de magasin de clés externe doté d'une connectivité au service de point de terminaison d'un VPC. Dans cet exemple, le magasin de clés externe est connecté à son proxy de magasin de clés externe.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
```

```
{
    "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}
```

Un `ConnectionState` dont la valeur est `Disconnected` indique que le magasin de clés externe n'a jamais été connecté ou qu'il a été intentionnellement déconnecté de son proxy de magasin de clés externe. Cependant, si les tentatives d'utilisation d'une clé KMS dans un magasin de clés externe connecté échouent, cela peut indiquer un problème avec le proxy du magasin de clés externe ou avec d'autres composants externes.

Si le `ConnectionState` du magasin de clés externe est `FAILED`, la réponse de `DescribeCustomKeyStores` inclut un élément `ConnectionErrorCode` qui explique la raison de l'erreur.

Par exemple, dans le résultat suivant, la `XKS_PROXY_TIMED_OUT` valeur indique qu'il est AWS KMS possible de se connecter au proxy de banque de clés externe, mais que la connexion a échoué car le proxy de banque de clés externe n'a pas répondu AWS KMS dans le délai imparti. Si ce code d'erreur de connexion s'affiche à plusieurs reprises, informez-en le fournisseur du proxy de votre magasin de clés externe. Pour obtenir de l'aide sur ce sujet et sur d'autres échecs de connexion, consultez Résoudre les problèmes liés aux magasins de clés externes.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
    "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
```

```
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "FAILED",
      "ConnectionErrorCode": "XKS_PROXY_TIMED_OUT",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}
```

Surveillez les magasins de clés externes

AWS KMS collecte des statistiques pour chaque interaction avec un magasin de clés externe et les publie sur votre CloudWatch compte. Ces métriques sont utilisées pour générer les graphiques dans la section de surveillance de la page détaillée pour chaque magasin de clés externe. La rubrique suivante explique comment utiliser les graphiques pour identifier et résoudre les problèmes de fonctionnement et de configuration affectant votre magasin de clés externe. Nous vous recommandons d'utiliser les CloudWatch métriques pour définir des alarmes qui vous avertissent lorsque votre magasin de clés externe ne fonctionne pas comme prévu. Pour plus d'informations, consultez la section Surveillance avec Amazon CloudWatch.

Rubriques

- Afficher les graphiques
- Interpréter les graphiques

Afficher les graphiques

Vous pouvez afficher les graphiques dans différents niveaux de détails. Par défaut, chaque graphique utilise une plage de temps de trois heures et une période d'agrégation de cinq minutes. Vous pouvez ajuster l'affichage graphique dans la console, mais vos modifications reviendront aux paramètres par défaut lorsque la page détaillée du magasin de clés externe sera fermée ou que le navigateur sera actualisé. Pour obtenir de l'aide sur CloudWatch la terminologie Amazon, consultez Amazon CloudWatch Concepts.

Afficher les détails des points de données

Les données de chaque graphique sont collectées par les métriques AWS KMS. Pour afficher plus d'informations sur un point de données spécifique, placez le pointeur de la souris sur le point de données du graphique linéaire. Cela affichera une fenêtre contextuelle contenant plus d'informations sur la métrique dont le graphique est issu. Chaque élément de la liste affiche la valeur de dimension enregistrée à ce point de données. La fenêtre contextuelle affiche une valeur nulle (–) si aucune donnée de métrique n'est disponible pour la valeur de dimension à ce point de données. Certains graphiques enregistrent plusieurs dimensions et valeurs pour un seul point de données. D'autres graphiques, tels que le graphique de fiabilité, utilisent les données collectées par la métrique pour calculer une valeur unique. Chaque élément de la liste est associé à une couleur de graphique linéaire différente.

Modifier la plage de temps

Pour modifier la plage de temps, sélectionnez l'une des plages de temps prédéfinies dans le coin supérieur droit de la section de surveillance. Les plages de temps prédéfinies s'étendent de 1 heure à 1 semaine (1 h, 3 h, 12 h, 1 j, 3 j, ou 1 sem). Cela permet d'ajuster la plage de temps pour tous les graphiques. Si vous souhaitez afficher un graphique spécifique dans un intervalle de temps différent, ou si vous souhaitez définir un intervalle de temps personnalisé, agrandissez-le ou affichez-le dans la CloudWatch console Amazon.

Zoom avant sur les graphiques

Vous pouvez utiliser la fonctionnalité de zoom de la mini-carte pour vous concentrer sur des sections de graphiques linéaires et des parties empilées des graphiques sans basculer entre les vues zoomée et dézoomée. Par exemple, vous pouvez utiliser la fonctionnalité de zoom de la mini-carte pour mettre l'accent sur un pic dans un graphique, de sorte que vous puissiez comparer le pic à d'autres graphiques de la section de surveillance provenant de la même chronologie.

1. Choisissez et faites glisser la zone du graphique sur laquelle vous souhaitez mettre l'accent, puis déposez.
2. Pour réinitialiser le zoom, choisissez l'icône Reset zoom (Réinitialiser le zoom), qui ressemble à une loupe avec un symbole moins (-) à l'intérieur.

Agrandir un graphique

Pour agrandir un graphique, sélectionnez l'icône de menu dans le coin supérieur droit d'un graphique individuel et choisissez Enlarge (Agrandir). Vous pouvez également sélectionner l'icône

d'agrandissement qui apparaît à côté de l'icône de menu lorsque vous passez la souris sur un graphique.

L'agrandissement d'un graphique vous permet de modifier davantage son affichage en spécifiant une période, une plage de temps personnalisée ou un intervalle d'actualisation différents. Ces modifications reviendront aux paramètres par défaut lorsque vous fermerez la vue agrandie.

Modifier la période

1. Choisissez le menu Period options (Options de période). Par défaut, ce menu affiche la valeur : 5 minutes.
2. Choisissez une période, les périodes prédéfinies s'étendent de 1 seconde à 30 jours.

   Par exemple, vous pouvez choisir une vue d'une minute, ce qui peut être utile lors d'un dépannage. Vous pouvez également choisir une vue moins détaillée, d'une heure par exemple. Cela peut être utile lors de l'affichage d'une plage de temps plus large (par exemple, 3 jours), afin de voir les tendances au fil du temps. Pour plus d'informations, consultez la section Périodes dans le guide de CloudWatch l'utilisateur Amazon.

Modifier la plage de temps ou le fuseau horaire

1. Sélectionnez l'une des plages de temps prédéfinies, qui s'étendent de 1 heure à 1 semaine (1 h, 3 h, 12 h, 1 j, 3 j, ou 1 sem). Vous pouvez également choisir Custom (Personnalisée) pour définir votre propre plage horaire.
2. Choisissez Custom (Personnalisée).

   a. Plage de temps : sélectionnez l'onglet Absolute (Absolue) dans le coin supérieur gauche de la boîte de dialogue. Utilisez le sélecteur de calendrier ou les champs de texte pour spécifier la plage de temps.

   b. Fuseau horaire : choisissez le menu déroulant dans le coin supérieur droit de la boîte de dialogue. Vous pouvez changer le fuseau horaire sur UTC ou Local time zone (Fuseau horaire local).
3. Une fois que vous avez spécifié une plage de temps, choisissez Apply (Appliquer).

Modifier la fréquence à laquelle les données de votre graphique sont actualisées

1. Dans le coin supérieur droit, choisissez le menu Refresh options (Options d'actualisation).
2. Choisissez un intervalle d'actualisation (Désactivé, 10 secondes, 1 minute, 2 minutes, 5 minutes ou 15 minutes).

Afficher les graphiques dans la CloudWatch console Amazon

Les graphiques de la section de surveillance sont dérivés de mesures prédéfinies AWS KMS publiées sur Amazon CloudWatch. Vous pouvez les ouvrir dans la CloudWatch console et les enregistrer dans des CloudWatch tableaux de bord. Si vous possédez plusieurs magasins de clés externes, vous pouvez ouvrir leurs graphiques respectifs CloudWatch et les enregistrer dans un tableau de bord unique pour comparer leur état de santé et leur utilisation.

Ajouter au CloudWatch tableau de bord

Sélectionnez Ajouter au tableau de bord dans le coin supérieur droit pour ajouter tous les graphiques à un tableau de CloudWatch bord Amazon. Vous pouvez utiliser un tableau de bord existant ou en créer un. Pour plus d'informations sur l'utilisation de ce tableau de bord pour créer des vues personnalisées des graphiques et des alarmes, consultez la section Utilisation CloudWatch des tableaux de bord Amazon dans le guide de CloudWatch l'utilisateur Amazon.

Afficher dans les CloudWatch métriques

Sélectionnez l'icône du menu dans le coin supérieur droit d'un graphique individuel et choisissez Afficher dans les métriques pour afficher ce graphique dans la CloudWatch console Amazon. Depuis la CloudWatch console, vous pouvez ajouter ce graphique unique à un tableau de bord et modifier les plages de temps, les périodes et les intervalles d'actualisation. Pour plus d'informations, consultez la section Représentation graphique des métriques dans le guide de CloudWatch l'utilisateur Amazon.

Interpréter les graphiques

AWS KMS fournit plusieurs graphiques pour surveiller l'état de votre magasin de clés externe dans la AWS KMS console. Ces graphiques sont automatiquement configurés et dérivés des métriques AWS KMS.

Les données de graphique sont collectées dans le cadre des appels que vous effectuez vers votre magasin de clés externe et vos clés externes. Vous pouvez voir des données remplir des graphiques pendant une période pendant laquelle vous n'avez passé aucun appel. Ces données proviennent des `GetHealthStatus` appels périodiques effectués en votre nom AWS KMS pour vérifier l'état de votre proxy de magasin de clés externe et de votre gestionnaire de clés externe. Si vos graphiques affichent le message No data available (Aucune donnée disponible), cela signifie qu'aucun appel n'a été enregistré au cours de cette période ou que votre magasin de clés externe est à l'état `DISCONNECTED`. Vous pouvez peut-être identifier l'heure à laquelle votre magasin de clés externe s'est déconnecté en ajustant votre affichage sur une plage de temps plus étendue.

Rubriques

Total requests (Nombre total de requêtes)

Nombre total de AWS KMS demandes reçues pour une banque de clés externe spécifique au cours d'une période donnée. Utilisez ce graphique pour déterminer si vous êtes exposé à un risque de limitation.

AWS KMS recommande que votre gestionnaire de clés externe soit capable de traiter jusqu'à 1 800 demandes d'opérations cryptographiques par seconde. Si vous approchez les 540 000 appels par période de cinq minutes, vous risquez d'être limité.

Vous pouvez surveiller le nombre de demandes d'opérations cryptographiques sur les clés KMS dans votre magasin de clés externe limité AWS KMS par la métrique. ExternalKeyStoreThrottle

Si vous recevez des erreurs `KMSInvalidStateException` très fréquentes avec un message expliquant que la requête a été rejetée « en raison d'un taux de requêtes très élevé », cela peut indiquer que votre gestionnaire de clés externe ou votre proxy de magasin de clés externe ne peuvent pas suivre le rythme du taux de requêtes actuel. Si possible, réduisez votre taux de requêtes. Vous pouvez également envisager de demander une diminution de la valeur de votre quota de requêtes du magasin de clés personnalisé. La diminution de cette valeur de quota peut augmenter la régulation, mais cela indique que les demandes excédentaires AWS KMS sont rejetées rapidement avant qu'elles ne soient envoyées à votre proxy de stockage de clés externe ou à votre gestionnaire de clés externe. Pour solliciter une réduction de quota, accédez au Centre AWS Support et créez une demande.

Le graphique du nombre total de requêtes est dérivé de la métrique XksProxyErrors, qui collecte des données sur les réponses fructueuses et infructueuses qu' AWS KMS reçoit de votre proxy de magasin de clés externe. Lorsque vous consultez un point de données spécifique, la fenêtre contextuelle affiche la valeur de la `CustomKeyStoreId` dimension ainsi que le nombre total de AWS KMS demandes enregistrées à ce point de données. Le `CustomKeyStoreId` sera toujours identique.

Fiabilité

Pourcentage de AWS KMS demandes pour lesquelles le proxy de stockage de clés externe a renvoyé une réponse réussie ou une erreur ne pouvant pas être réessayée. Utilisez ce graphique pour évaluer l'état opérationnel de votre proxy de magasin de clés externe.

Lorsque le graphique affiche une valeur inférieure à 100 %, il indique les cas où le proxy n'a pas répondu ou a répondu par une erreur récupérable. Cela peut indiquer des problèmes liés au réseau, une lenteur du proxy de magasin de clés externe ou du gestionnaire de clés externe, ou des bogues d'implémentation.

Si la requête inclut des informations d'identification erronées et que votre proxy répond par une exception `AuthenticationFailedException`, le graphique indiquera toujours une fiabilité de 100 %, car le proxy a identifié une valeur incorrecte dans la requête d'API de proxy de magasin de clés externe. Par conséquent, l'échec est prévisible. Si le pourcentage de votre graphique de fiabilité est de 100 %, cela signifie que le proxy de votre magasin de clés externe répond comme prévu. Si le graphique affiche une valeur inférieure à 100 %, le proxy a répondu par une erreur récupérable ou a expiré. Par exemple, si le proxy répond par une exception `ThrottlingException` en raison d'un taux de requêtes très élevé, il affichera un pourcentage de fiabilité inférieur, car le proxy n'a pas été en mesure d'identifier un problème spécifique dans la requête qui a provoqué son échec. En effet, les erreurs récupérables sont probablement des problèmes transitoires qui peuvent être résolus en répétant la requête.

Les réponses d'erreur suivantes réduiront le pourcentage de fiabilité. Vous pouvez utiliser le graphique Les cinq principales exceptions et la métrique XksProxyErrors pour surveiller davantage la fréquence à laquelle votre proxy renvoie chaque erreur récupérable.

- `InternalException`
- `DependencyTimeoutException`
- `ThrottlingException`
- `XksProxyUnreachableException`

Le graphique de fiabilité est dérivé de la XksProxyErrors métrique, qui collecte des données sur les réponses positives et infructueuses AWS KMS reçues de votre proxy de stockage de clés externe. Le pourcentage de fiabilité ne diminue que si la réponse a une valeur `ErrorType` égale à `Retryable`. Lorsque vous consultez un point de données spécifique, la fenêtre contextuelle affiche la valeur de la `CustomKeyStoreId` dimension ainsi que le pourcentage de fiabilité pour les AWS KMS demandes enregistrées à ce point de données. Le `CustomKeyStoreId` sera toujours identique.

Nous vous recommandons d'utiliser cette XksProxyErrors métrique pour créer une CloudWatch alarme qui vous avertit des problèmes potentiels liés au réseau en vous alertant lorsque plus de cinq erreurs réessayables sont enregistrées en une minute. Pour de plus amples informations, veuillez consulter Créez une alarme pour les erreurs réessayables.

Latence

Le nombre de millisecondes nécessaires à un proxy de stockage de clés externe pour répondre à une demande. AWS KMS Utilisez ce graphique pour évaluer les performances de votre proxy de magasin de clés externe et de votre gestionnaire de clés externe.

AWS KMS attend du proxy de stockage de clés externe qu'il réponde à chaque demande dans un délai de 250 millisecondes. En cas d'expiration du délai d'attente du réseau, la demande AWS KMS sera réessayée une fois. Si le proxy échoue une seconde fois, la latence enregistrée est le délai d'expiration combiné pour les deux tentatives de requête et le graphique affichera environ 500 millisecondes. Dans tous les autres cas où le proxy ne répond pas dans la limite du délai d'expiration de 250 millisecondes, la latence enregistrée est de 250 millisecondes. Si le proxy expire fréquemment lors des opérations de chiffrement et de déchiffrement, consultez votre administrateur proxy externe. Pour obtenir de l'aide afin de résoudre les problèmes de latence, veuillez consulter la rubrique Erreurs de latence et de délai d'expiration.

Les réponses lentes peuvent également indiquer que votre gestionnaire de clés externe ne peut pas gérer le trafic de demandes actuel. AWS KMS recommande que votre gestionnaire de clés externe soit capable de traiter jusqu'à 1 800 demandes d'opérations cryptographiques par seconde. Si votre gestionnaire de clés externe ne peut pas gérer le taux de 1 800 requêtes par seconde, pensez à demander une diminution de votre quota de requêtes de clés KMS dans un magasin de clés personnalisé. Les requêtes d'opérations cryptographiques utilisant les clés KMS de votre magasin de clés externe échoueront rapidement, avec une exception de limitation, au lieu d'être traitées puis rejetées par le proxy de votre magasin de clés externe ou le gestionnaire de clés externe.

Le graphique de latence est dérivé de la métrique XksProxyLatency. Lorsque vous consultez un point de données spécifique, la fenêtre contextuelle affiche les valeurs des dimensions `KmsOperation` et `XksOperation` correspondantes, ainsi que la latence moyenne enregistrée pour les opérations sur ce point de données. Les éléments de la liste sont classés de la latence la plus élevée à la plus faible.

Nous vous recommandons d'utiliser cette XksProxyLatency métrique pour créer une CloudWatch alarme qui vous avertira lorsque votre latence approche de la limite de temporisation. Pour de plus amples informations, veuillez consulter Création d'une alarme pour l'expiration du délai de réponse.

Les cinq principales exceptions

Les cinq principales exceptions en cas d'échec des opérations cryptographiques et de gestion au cours d'une période donnée. Utilisez ce graphique pour suivre les erreurs les plus fréquentes, afin de prioriser votre effort d'ingénierie.

Ce décompte inclut les exceptions AWS KMS reçues du proxy de stockage de clés externe et celles renvoyées `XksProxyUnreachableException` en interne lorsqu'il ne peut pas établir de communication avec le proxy de stockage de clés externe. AWS KMS

Des taux élevés d'erreurs récupérables peuvent indiquer des erreurs réseau, tandis que des taux élevés d'erreurs non récupérables peuvent indiquer un problème de configuration de votre magasin de clés externe. Par exemple, un pic `AuthenticationFailedExceptions` indique une différence entre les informations d'authentification configurées dans le proxy de stockage de clés externe AWS KMS et celles configurées dans le proxy de stockage de clés externe. Pour consulter la configuration de votre magasin de clés externe, veuillez consulter la rubrique Afficher les magasins de clés externes. Pour modifier les paramètres de votre clé externe, veuillez consulter la rubrique Modifier les propriétés du magasin de clés externe.

Les exceptions AWS KMS reçues du proxy de stockage de clés externe sont différentes des exceptions renvoyées AWS KMS en cas d'échec d'une opération. AWS KMS les opérations cryptographiques renvoient un `KMSInvalidStateException` pour toutes les défaillances liées à la configuration externe ou à l'état de connexion du magasin de clés externe. Pour identifier le problème, utilisez le texte du message d'erreur qui l'accompagne.

Le tableau suivant indique les exceptions qui peuvent apparaître dans le graphique des 5 principales exceptions et les exceptions correspondantes qui vous sont AWS KMS renvoyées.

Error type (Type d'erreur)	Exception affichée dans le graphique	Exception qui vous AWS KMS est revenue
Non récupérable	**`AccessDeniedException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Problèmes d'autorisation du proxy.	**`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.

Error type (Type d'erreur)	Exception affichée dans le graphique	Exception qui vous AWS KMS est revenue
Non récupérable	**`AuthenticationFailedException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs liées aux informations d'identification pour l'authentification.	**`XksProxyIncorrectAuthenticationCredentialException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Récupérable	**`DependencyTimeoutException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs de latence et de délai d'expiration.	**`XksProxyUriUnreachableException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.

Error type (Type d'erreur)	Exception affichée dans le graphique	Exception qui vous AWS KMS est revenue
Récupérable	**`InternalException`** Le proxy de magasin de clés externe a rejeté la requête, car il ne peut pas communiquer avec le gestionnaire de clés externe. Vérifiez que la configuration du proxy de magasin de clés externe est correcte et que le gestionnaire de clés externe est disponible.	**`XksProxyInvalidResponseException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Non récupérable	**`InvalidCiphertextException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs de déchiffrement.	**`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Non récupérable	**`InvalidKeyUsageException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs d'opérations cryptographiques pour la clé externe.	**`XksKeyInvalidConfigurationException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.

Error type (Type d'erreur)	Exception affichée dans le graphique	Exception qui vous AWS KMS est revenue
Non récupérable	**`InvalidStateException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs d'opérations cryptographiques pour la clé externe.	**`XksKeyInvalidConfigurationException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Non récupérable	**`InvalidUriPathException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs de configuration générale.	**`XksProxyInvalidConfigurationException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Non récupérable	**`KeyNotFoundException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs liées aux clés externes.	**`XksKeyNotFoundException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.

Error type (Type d'erreur)	Exception affichée dans le graphique	Exception qui vous AWS KMS est revenue
Récupérable	**`ThrottlingException`** Le proxy de magasin de clés externe a rejeté la requête en raison d'un taux de requêtes très élevé. Réduisez la fréquence de vos appels utilisant des clés KMS dans ce magasin de clés externe.	**`XksProxyUriUnreachableException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Non récupérable	**`UnsupportedOperationException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Erreurs d'opérations cryptographiques pour la clé externe.	**`XksKeyInvalidResponseException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.

Error type (Type d'erreur)	Exception affichée dans le graphique	Exception qui vous AWS KMS est revenue
Non récupérable	**`ValidationException`** Pour bénéficier d'une aide à la résolution des problèmes, consultez Problèmes liés aux proxys.	**`XksProxyInvalidResponseException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.
Récupérable	**`XksProxyUnreachableException`** Si cette erreur s'affiche à plusieurs reprises, vérifiez que le proxy de votre magasin de clés externe est actif et connecté au réseau, et que son chemin d'URI et son URI de point de terminaison ou le nom de service VPC sont corrects dans votre magasin de clés externe.	**`XksProxyUriUnreachableException`** dans la réponse aux opérations `CreateCustomKeyStore` et `UpdateCustomKeyStore` . **`CustomKeyStoreInvalidStateException`** dans la réponse aux opérations `CreateKey` . **`KMSInvalidStateException`** dans la réponse aux opérations cryptographiques.

Le graphique des cinq principales exceptions est dérivé de la métrique XksProxyErrors. Lorsque vous consultez un point de données spécifique, la fenêtre contextuelle affiche la valeur de la

dimension `ExceptionName` ainsi que le nombre de fois que l'exception a été enregistrée à ce point de données. Les cinq éléments de la liste sont classés de l'exception la plus fréquente à la moins fréquente.

Nous vous recommandons d'utiliser cette XksProxyErrors métrique pour créer une CloudWatch alarme qui vous avertit des problèmes de configuration potentiels en vous alertant lorsque plus de cinq erreurs non réessayables sont enregistrées en une minute. Pour de plus amples informations, veuillez consulter Créez une alarme pour les erreurs non réessayables.

Nombre de jours avant l'expiration du certificat

Nombre de jours avant l'expiration du certificat TLS de votre point de terminaison du proxy de magasin de clés externe (`XksProxyUriEndpoint`). Utilisez ce graphique pour surveiller l'expiration imminente de votre certificat TLS.

Lorsque le certificat expire, AWS KMS impossible de communiquer avec le proxy de stockage de clés externe. Toutes les données protégées par des clés KMS dans votre magasin de clés externe deviennent inaccessibles jusqu'à ce que vous renouveliez le certificat.

Le graphique du nombre de jours avant l'expiration du certificat est dérivé de la métrique XksProxyCertificateDaysToExpire. Nous vous recommandons vivement d'utiliser cette métrique pour créer une CloudWatch alarme qui vous avertira de l'expiration prochaine. L'expiration du certificat peut vous empêcher d'accéder à vos ressources chiffrées. Réglez l'alerte pour donner à votre organisation le temps de renouveler le certificat avant qu'il n'expire. Pour de plus amples informations, veuillez consulter Création d'une alarme pour l'expiration du certificat.

Connecter et déconnecter les magasins de clés externes

Les nouveaux magasins de clés externes ne sont pas connectés. Pour créer et utiliser AWS KMS keys dans votre magasin de clés externe, vous devez connecter votre magasin de clés externe à son proxy de magasin de clés externe. Vous pouvez connecter et déconnecter votre magasin de clés externe à tout moment, et afficher son état de connexion.

Lorsque votre magasin de clés externe est déconnecté, vous AWS KMS ne pouvez pas communiquer avec votre proxy de magasin de clés externe. Par conséquent, vous pouvez afficher et gérer votre magasin de clés externe et ses clés KMS existantes. Toutefois, vous ne pouvez pas créer de clés KMS dans votre magasin de clés externe, ni utiliser ses clés KMS dans des opérations cryptographiques. Il se peut que vous deviez déconnecter votre magasin de clés externe à un moment donné, par exemple lorsque vous modifiez ses propriétés, mais planifiez cette action en

conséquence. La déconnexion du magasin de clés peut perturber le fonctionnement des AWS services qui utilisent ses clés KMS.

Vous n'avez pas besoin de connecter votre magasin de clés externe. Vous pouvez conserver un magasin de clés externe dans un état déconnecté indéfiniment et le connecter uniquement lorsque vous avez besoin de l'utiliser. Cependant, vous pouvez tester la connexion régulièrement pour vérifier que les paramètres sont corrects et que le magasin peut être connecté.

Lorsque vous déconnectez un magasin de clés personnalisé, les clés KMS du magasin de clés deviennent immédiatement inutilisables (sous réserve d'une éventuelle cohérence). Toutefois, les ressources chiffrées à l'aide de clés de données protégées par la clé KMS ne sont pas affectées tant que la clé KMS n'est pas réutilisée, par exemple pour déchiffrer la clé de données. Ce problème affecte les Services AWS, dont beaucoup utilisent des clés de données pour protéger vos ressources. Pour plus de détails, consultez Comment les clés KMS inutilisables affectent les clés de données.

> **Note**
>
> Les magasins de clés externes sont à l'état `DISCONNECTED` uniquement lorsque le magasin de clés n'a jamais été connecté ou que vous le déconnectez explicitement. Un état `CONNECTED` n'indique pas que le magasin de clés externe ou ses composants de support fonctionnent efficacement. Pour plus d'informations sur les performances des composants de votre magasin de clés externe, veuillez consulter les graphiques de la section Monitoring (Surveillance) de la page détaillée de chaque magasin de clés externe. Pour plus de détails, consultez Surveillez les magasins de clés externes.
>
> Votre gestionnaire de clés externe peut fournir des méthodes supplémentaires pour arrêter et redémarrer la communication entre votre magasin de clés AWS KMS externe et votre proxy de magasin de clés externe, ou entre votre proxy de magasin de clés externe et le gestionnaire de clés externe. Pour en savoir plus, veuillez consulter la documentation de votre gestionnaire de clés externe.

Rubriques

État de connexion

La connexion et la déconnexion modifient l'état de connexion de votre magasin de clés personnalisé. Les valeurs d'état de connexion sont les mêmes pour les magasins de AWS CloudHSM clés et les magasins de clés externes.

Pour afficher l'état de connexion de votre magasin de clés personnalisé, utilisez l'DescribeCustomKeyStoresopération ou la AWS KMS console. L'état de la connexion apparaît dans chaque tableau de magasin de clés personnalisé, dans la section General configuration (Configuration générale) de la page détaillée de chaque magasin de clés personnalisé et dans l'onglet Cryptographic configuration (Configuration cryptographique) des clés KMS d'un magasin de clés personnalisé. Pour plus d'informations, consultez Afficher un magasin AWS CloudHSM de clés et Afficher les magasins de clés externes.

Un magasin de clés personnalisé peut avoir l'un des états de connexion suivants :

- `CONNECTED` : le magasin de clés personnalisé est connecté à son magasin de clés de sauvegarde. Vous pouvez créer et utiliser les clés KMS dans le magasin de clés personnalisé.

  Le magasin de clés de sauvegarde d'un magasin de AWS CloudHSM clés est son AWS CloudHSM cluster associé. Le magasin de clés de sauvegarde d'un magasin de clés externe est constitué du proxy de magasin de clés externe et du gestionnaire de clés externe qu'il prend en charge.

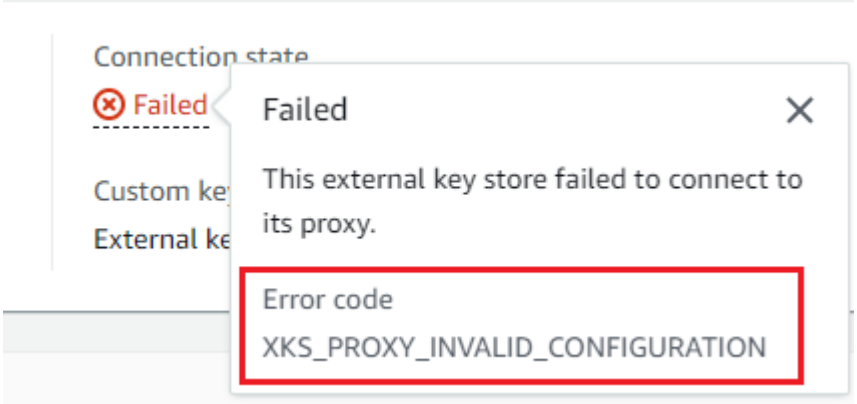
  Un état CONNECTÉ signifie que la connexion s'est établie et que le magasin de clés personnalisé n'a pas été déconnecté intentionnellement. Cela n'indique pas que la connexion fonctionne correctement. Pour plus d'informations sur l'état du AWS CloudHSM cluster associé à votre magasin de AWS CloudHSM clés, consultez la section Obtenir CloudWatch des métriques AWS CloudHSM dans le guide de AWS CloudHSM l'utilisateur. Pour plus d'informations sur l'état et le fonctionnement de votre magasin de clés externe, veuillez consulter les graphiques de la section Monitoring (Surveillance) de la page détaillée de chaque magasin de clés externe. Pour plus de détails, consultez Surveillez les magasins de clés externes.
- `CONNECTING` : le processus de connexion d'un magasin de clés personnalisé est en cours. Il s'agit d'un état transitoire.
- `DISCONNECTED`: Le magasin de clés personnalisé n'a jamais été connecté à son support, ou il a été déconnecté intentionnellement à l'aide de la AWS KMS console ou de l'DisconnectCustomKeyStoreopération.
- `DISCONNECTING` : le processus de déconnexion d'un magasin de clés personnalisé est en cours. Il s'agit d'un état transitoire.

- `FAILED` : une tentative de connexion du magasin de clés personnalisé a échoué. Le « `ConnectionErrorCode` in » de la DescribeCustomKeyStoresréponse indique le problème.

Pour connecter un magasin de clés personnalisé, son état de connexion doit être `DISCONNECTED`. Si l'état de la connexion est `FAILED`, utilisez le `ConnectionErrorCode` pour identifier et résoudre le problème. Déconnectez ensuite le magasin de clés personnalisé avant d'essayer de le connecter à nouveau. Pour obtenir de l'aide concernant les connexions ayant échoué, veuillez consulter Erreurs de connexion au magasin de clés externe. Pour obtenir de l'aide afin de répondre à un code d'erreur de connexion, veuillez consulter la rubrique Codes d'erreur de connexion pour les magasins de clés externes.

Pour consulter le code d'erreur de connexion, procédez comme suit :

- Dans la DescribeCustomKeyStoresréponse, visualisez la valeur de l'`ConnectionErrorCode`élément. Cet élément apparaît dans la réponse de `DescribeCustomKeyStores` uniquement lorsque le `ConnectionState` est `FAILED`.
- Pour afficher le code d'erreur de connexion dans la AWS KMS console, sur la page détaillée du magasin de clés externe, passez le curseur sur la valeur Échec.



Connecter un magasin de clés externe

Lorsque votre magasin de clés externe est connecté à son proxy de magasin de clés externe, vous pouvez créer des clés KMS dans votre magasin de clés externe et utiliser les clés KMS existantes dans les opérations cryptographiques.

Le processus qui connecte un magasin de clés externe à son proxy de magasin de clés externe varie en fonction de la connectivité du magasin de clés externe.

- Lorsque vous connectez un magasin de clés externe connecté à un point de terminaison public, AWS KMS envoie une GetHealthStatus demande au proxy du magasin de clés externe pour valider le point de terminaison de l'URI du proxy, le chemin de l'URI du proxy et les informations d'authentification du proxy. Une réponse positive du proxy confirme que le point de terminaison d'URI de proxy et le chemin d'URI de proxy sont exacts et accessibles, et que le proxy a authentifié la requête signée à l'aide des informations d'identification pour l'authentification du proxy pour le magasin de clés externe.
- Lorsque vous connectez un magasin de clés externe connecté au service de point de terminaison VPC à son proxy de magasin de clés externe, procédez AWS KMS comme suit :
  - Il confirme que le domaine pour le nom DNS privé spécifié dans le point de terminaison d'URI de proxy est vérifié.
  - Crée un point de terminaison d'interface entre un AWS KMS VPC et votre service de point de terminaison VPC.
  - Il crée une zone hébergée privée pour le nom DNS privé spécifié dans le point de terminaison d'URI de proxy.
  - Envoie une GetHealthStatusdemande au proxy de stockage de clés externe. Une réponse positive du proxy confirme que le point de terminaison d'URI de proxy et le chemin d'URI de proxy sont exacts et accessibles, et que le proxy a authentifié la requête signée à l'aide des informations d'identification pour l'authentification du proxy pour le magasin de clés externe.

L'opération de connexion lance le processus de connexion de votre magasin de clés personnalisé, mais la connexion d'un magasin de clés externe à son proxy externe prend environ cinq minutes. Une réponse positive à l'opération de connexion n'indique pas que le magasin de clés externe est connecté. Pour confirmer que la connexion a été établie, utilisez la AWS KMS console ou l'DescribeCustomKeyStoresopération pour afficher l'état de connexion de votre magasin de clés externe.

Lorsque l'état de connexion est `FAILED` atteint, un code d'erreur de connexion s'affiche dans la AWS KMS console et est ajouté à la `DescribeCustomKeyStore` réponse. Pour obtenir de l'aide sur l'interprétation des codes d'erreur de connexion, veuillez consulter la rubrique Codes d'erreur de connexion pour les magasins de clés externes.

Connectez-vous et reconnectez-vous à votre magasin de clés externe

Vous pouvez connecter ou reconnecter votre banque de clés externe dans la AWS KMS console ou en utilisant l'ConnectCustomKeyStoreopération.

Utilisation de la AWS KMS console

Vous pouvez utiliser la AWS KMS console pour connecter un magasin de clés externe à son proxy de magasin de clés externe.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Choisissez la ligne du magasin de clés externe que vous souhaitez connecter.

   Si l'état de connexion du magasin de clés externe est FAILED (ÉCHEC), vous devez déconnecter le magasin de clés externe avant de le connecter.
5. Dans le menu Key store actions (Actions de magasin de clés), choisissez Connect (Connecter).

Le processus de connexion prend en général environ cinq minutes. Lorsque l'opération est terminée, l'état de connexion passe à CONNECTED (CONNECTÉ).

Si l'état de connexion est Failed (Échec), passez la souris sur l'état de la connexion pour voir le code d'erreur de connexion, qui explique la cause de l'erreur. Pour obtenir de l'aide afin de répondre à un code d'erreur de connexion, veuillez consulter la rubrique Codes d'erreur de connexion pour les magasins de clés externes. Pour connecter un magasin de clés externe dont l'état de connexion est Failed (Échec), vous devez d'abord déconnecter le magasin de clés personnalisé.

Utilisation de l' AWS KMS API

Pour connecter un magasin de clés externe déconnecté, utilisez l'ConnectCustomKeyStoreopération.

Avant la connexion, l'état de connexion du magasin de clés externe doit être `DISCONNECTED`. Si l'état actuel de la connexion est `FAILED`, déconnectez le magasin de clés externe, puis connectez-le.

Le processus de connexion prend environ cinq minutes. À moins qu'elle n'échoue rapidement, `ConnectCustomKeyStore` renvoie une réponse HTTP 200 et un objet JSON sans propriétés. Cependant, cette réponse initiale n'indique pas que la connexion a abouti. Pour déterminer si le magasin de clés externe est connecté, consultez l'état de la connexion dans la DescribeCustomKeyStoresréponse.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Pour identifier le magasin de clés externe, utilisez son ID du magasin de clés personnalisé. Vous pouvez trouver l'ID sur la page des stockages de clés personnalisés de la console ou en utilisant l'DescribeCustomKeyStoresopération. Avant d'exécuter cet exemple, remplacez l'ID de l'exemple par un ID valide.

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

L'opération `ConnectCustomKeyStore` ne renvoie pas de `ConnectionState` dans sa réponse. Pour vérifier que le magasin de clés externe est connecté, utilisez l'DescribeCustomKeyStoresopération. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Toutefois, vous pouvez utiliser le paramètre `CustomKeyStoreName`ou `CustomKeyStoreId` (mais pas les deux) pour limiter la réponse à des magasins de clés personnalisés en particulier. Une valeur de `ConnectionState` égale à `CONNECTED` indique que le magasin de clés externe est connecté à son proxy de magasin de clés externe.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
    "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}
```

Si la valeur de `ConnectionState` dans la réponse de `DescribeCustomKeyStores` est `FAILED`, l'élément `ConnectionErrorCode` indique la raison de l'échec.

Dans l'exemple suivant, la `XKS_VPC_ENDPOINT_SERVICE_NOT_FOUND` valeur de `ConnectionErrorCode` indique que AWS KMS le service de point de terminaison VPC qu'il utilise pour communiquer avec le proxy de banque de clés externe est introuvable. Vérifiez que `XksProxyVpcEndpointServiceName` c'est correct, que le principal de AWS KMS service est un principal autorisé sur le service de point de terminaison Amazon VPC et que le service de point de terminaison VPC n'exige pas l'acceptation des demandes de connexion. Pour obtenir de l'aide afin de répondre à un code d'erreur de connexion, veuillez consulter la rubrique Codes d'erreur de connexion pour les magasins de clés externes.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
    "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "FAILED",
      "ConnectionErrorCode": "XKS_VPC_ENDPOINT_SERVICE_NOT_FOUND",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}
```

Déconnecter un magasin de clés externe

Lorsque vous déconnectez un magasin de clés externe doté d'une connectivité au service de point de terminaison d'un VPC de son proxy de magasin de clés externe, AWS KMS supprime son point de terminaison d'interface vers le service de point de terminaison d'un VPC et supprime l'infrastructure réseau qu'il a créée pour prendre en charge la connexion. Aucun processus équivalent n'est requis pour les magasins de clés externes disposant d'une connectivité au point de terminaison public. Cette

action n'affecte pas le service de point de terminaison d'un VPC ni aucun de ses composants de support, et elle n'affecte pas le proxy de magasin de clés externe ni aucun composant externe.

Lorsque le magasin de clés externe est déconnecté, AWS KMS n'envoie aucune demande au proxy du magasin de clés externe. L'état de connexion du magasin de clés externe est `DISCONNECTED`. Les clés KMS du magasin de clés externe déconnecté sont dans un état de clé `UNAVAILABLE` (sauf si elles sont en attente de suppression), ce qui signifie qu'elles ne peuvent pas être utilisées dans des opérations cryptographiques. Toutefois, vous pouvez toujours consulter et gérer votre magasin de clés externe et ses clés KMS existantes.

L'état déconnecté est conçu pour être temporaire et réversible. Vous pouvez reconnecter votre magasin de clés externe à tout moment. En général, aucune reconfiguration n'est nécessaire. Cependant, si des propriétés du proxy de magasin de clés externe associé ont changé pendant sa déconnexion, par exemple la rotation de ses informations d'identification pour l'authentification du proxy, vous devez modifier les paramètres du magasin de clés externe avant de le reconnecter.

> **Note**
>
> Même si un magasin de clés personnalisé est déconnecté, toutes les tentatives de création de clés KMS dans le magasin de clés personnalisé ou d'utilisation de clés KMS existantes dans les opérations de chiffrement échouent. Cette action peut empêcher les utilisateurs de stocker des données sensibles et d'y accéder.

Pour mieux estimer l'effet de la déconnexion de votre magasin de clés externe, identifiez les clés KMS du magasin de clés externe et déterminez leur utilisation antérieure.

Vous pouvez déconnecter le magasin de clés externe pour des raisons telles que les suivantes :

- Pour modifier ses propriétés. Vous pouvez modifier le nom du magasin de clés personnalisé, le chemin d'URI de proxy et les informations d'identification pour l'authentification du proxy lorsque le magasin de clés externe est connecté. Toutefois, pour modifier le type de connectivité du proxy, le point de terminaison de l'URI de proxy ou le nom du service de point de terminaison d'un VPC, vous devez d'abord déconnecter le magasin de clés externe. Pour plus de détails, consultez Modifier les propriétés du magasin de clés externe.
- Pour arrêter toute communication entre AWS KMS et le proxy de stockage de clés externe. Vous pouvez également arrêter la communication entre AWS KMS et votre proxy en désactivant votre point de terminaison ou le service de point de terminaison VPC. En outre, votre proxy de stockage de clés externe ou votre logiciel de gestion de clés peuvent fournir des mécanismes

supplémentaires pour AWS KMS empêcher la communication avec le proxy ou pour empêcher le proxy d'accéder à votre gestionnaire de clés externe.

- Pour désactiver toutes les clés KMS du magasin de clés externe. Vous pouvez désactiver et réactiver les clés KMS dans un magasin de clés externe à l'aide de la AWS KMS console ou de l'DisableKeyopération. Ces opérations se déroulent rapidement (sous réserve d'une éventuelle cohérence), mais elles n'agissent que sur une seule clé KMS à la fois. La déconnexion du magasin de clés externe fait passer l'état de clé de toutes les clés KMS dans le magasin de clés externe à `Unavailable`, ce qui empêche leur utilisation dans les opérations cryptographiques.
- Pour réparer un échec de tentative de connexion. Si une tentative de connexion d'un magasin de clés externe échoue (l'état de connexion du magasin de clés personnalisé est `FAILED`), vous devez déconnecter le magasin de clés externe avant d'essayer de le connecter à nouveau.

Déconnectez votre magasin de clés externe

Vous pouvez déconnecter votre porte-clés externe dans la AWS KMS console ou en utilisant cette DisconnectCustomKeyStoreopération.

Utilisation de la AWS KMS console

Vous pouvez utiliser la AWS KMS console pour connecter un magasin de clés externe à son proxy de magasin de clés externe. Ce processus prend environ cinq minutes.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.
2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Choisissez la ligne du magasin de clés externe que vous souhaitez déconnecter.
5. Dans le menu Key store actions (Actions de magasin de clés), choisissez Disconnect (Déconnecter).

Une fois l'opération terminée, l'état de la connexion passe de DISCONNECTING à DISCONNECTED. Si l'opération échoue, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez Erreurs de connexion au magasin de clés externe.

Utilisation de l' AWS KMS API

Pour déconnecter un magasin de clés externe connecté, utilisez l'DisconnectCustomKeyStoreopération. Si l'opération aboutit, AWS KMS renvoie une réponse HTTP 200 et un objet JSON sans propriétés. Le processus prend environ cinq minutes. Pour connaître l'état de connexion du magasin de clés externe, utilisez l'DescribeCustomKeyStoresopération.

Les exemples de cette section utilisent la AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge.

Cet exemple déconnecte un magasin de clés externe doté d'une connectivité au service de point de terminaison d'un VPC. Avant d'exécuter cet exemple, remplacez l'exemple d'ID de magasin de clés personnalisé par un ID valide.

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Pour vérifier que le magasin de clés externe est déconnecté, utilisez l'DescribeCustomKeyStoresopération. Par défaut, cette opération renvoie tous les magasins de clés personnalisés de vos compte et région. Toutefois, vous pouvez utiliser le paramètre `CustomKeyStoreName` ou `CustomKeyStoreId` (mais pas les deux) pour limiter la réponse à des magasins de clés personnalisés en particulier. La valeur de `ConnectionState` égale à `DISCONNECTED` indique que cet exemple de magasin de clés externe n'est plus connecté à son proxy de magasin de clés externe.

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
    "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "DISCONNECTED",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
```

```
  ]
}
```

Supprimer un magasin de clés externe

Lorsque vous supprimez un magasin de clés externe, toutes AWS KMS les métadonnées le concernant sont supprimées AWS KMS, y compris les informations relatives à son proxy de magasin de clés externe. Cette opération n'affecte pas le proxy de stockage de clés externe, le gestionnaire de clés externe, les clés externes, ni les AWS ressources que vous avez créées pour prendre en charge le magasin de clés externe, comme un Amazon VPC ou un service de point de terminaison VPC.

Avant de supprimer un magasin de clés externe, vous devez supprimer toutes les clés KMS du magasin de clés et déconnecter le magasin de clés de son proxy de magasin de clés externe. Dans le cas contraire, les tentatives de suppression du magasin de clés échouent.

La suppression d'un magasin de clés externe est irréversible, mais vous pouvez créer un autre magasin de clés externe et l'associer au même proxy de magasin de clés externe et au même gestionnaire de clés externe. Toutefois, vous ne pouvez pas recréer les clés KMS de chiffrement symétriques dans le magasin de clés externe, même si vous avez accès au même contenu de clé externe. AWS KMS inclut des métadonnées dans le texte chiffré symétrique propre à chaque clé KMS. Cette fonctionnalité de sécurité garantit que seule la clé KMS qui a chiffré des données peut les déchiffrer.

Au lieu de supprimer le magasin de clés externe, pensez à le déconnecter. Lorsqu'un magasin de clés externe est déconnecté, vous pouvez gérer le magasin de clés externe et le sien, AWS KMS keys mais vous ne pouvez pas créer ou utiliser de clés KMS dans le magasin de clés externe. Vous pouvez reconnecter le magasin de clés externe à tout moment et recommencer à utiliser ses clés KMS pour chiffrer et déchiffrer des données. Aucuns frais ne s'appliquent à un proxy de magasin de clés externe déconnecté ou lorsque ses clés KMS sont indisponibles.

Vous pouvez supprimer votre magasin de clés externe dans la AWS KMS console ou en utilisant cette DeleteCustomKeyStoreopération.

Utilisation de la AWS KMS console

Vous pouvez utiliser la AWS KMS console pour supprimer un magasin de clés externe.

1. Connectez-vous à la console AWS Key Management Service (AWS KMS) AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/kms.

2. Pour modifier le Région AWS, utilisez le sélecteur de région dans le coin supérieur droit de la page.
3. Dans le panneau de navigation, sélectionnez Custom key stores (Magasins de clés personnalisés), External key stores (Magasins de clés externes).
4. Recherchez la ligne qui représente le magasin de clés externe que vous souhaitez supprimer. Si Connection state (État de connexion) du magasin de clés externe n'est pas DISCONNECTED (DÉCONNECTÉ), vous devez déconnecter le magasin de clés externe avant de le supprimer.
5. Dans le menu Key store actions (Actions de magasin de clés), choisissez Delete (Supprimer).

Une fois l'opération terminée, un message de réussite s'affiche et le magasin de clés externe n'apparaît plus dans la liste des magasins de clés. Si l'opération échoue, un message d'erreur s'affiche qui décrit le problème et fournit une aide pour le résoudre. Si vous avez besoin d'aide supplémentaire, consultez Résoudre les problèmes liés aux magasins de clés externes.

Utilisation de l' AWS KMS API

Pour supprimer un magasin de clés externe, utilisez l'DeleteCustomKeyStoreopération. Si l'opération aboutit, AWS KMS renvoie une réponse HTTP 200 et un objet JSON sans propriétés.

Pour commencer, déconnectez le magasin de clés externe. Avant d'exécuter la commande, remplacez l'exemple d'ID de magasin de clés personnalisé par un ID valide.

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Une fois le magasin de clés externe déconnecté, vous pouvez utiliser DeleteCustomKeyStorecette opération pour le supprimer.

```
$ aws kms delete-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

Pour confirmer que le magasin de clés externe est supprimé, utilisez l'DescribeCustomKeyStoresopération.

```
$ aws kms describe-custom-key-stores

{
    "CustomKeyStores": []
}
```

Si vous spécifiez un nom ou un identifiant de banque de clés personnalisé qui n'existe plus, AWS KMS renvoie une `CustomKeyStoreNotFoundException` exception.

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0

An error occurred (CustomKeyStoreNotFoundException) when calling the
 DescribeCustomKeyStore operation:
```

Résoudre les problèmes liés aux magasins de clés externes

La résolution de la plupart des problèmes liés aux banques de clés externes est indiquée par le message d'erreur qui AWS KMS s'affiche à chaque exception ou par le code d'erreur de connexion qui s' AWS KMS affiche lorsqu'une tentative de connexion de la banque de clés externe à son proxy de banque de clés externe échoue. Toutefois, certains problèmes sont un peu plus complexes.

Lorsque vous diagnostiquez un problème lié à un magasin de clés externe, commencez par en rechercher la cause. Cela réduira le champ des possibles et rendra votre dépannage plus efficace.

- AWS KMS — Le problème peut être interne AWS KMS, par exemple une valeur incorrecte dans la configuration de votre magasin de clés externe.
- Externe : le problème peut provenir de l'extérieur AWS KMS, notamment des problèmes liés à la configuration ou au fonctionnement du proxy de stockage de clés externe, du gestionnaire de clés externe, des clés externes ou du service de point de terminaison VPC.
- Mise en réseau : il peut s'agir d'un problème de connectivité ou de mise en réseau, tel qu'un problème lié à votre point de terminaison proxy, à votre port, à votre pile d'adresses IP ou à votre nom ou domaine DNS privé.

> **Note**
>
> Lorsque les opérations de gestion sur des magasins de clés externes échouent, elles génèrent plusieurs exceptions différentes. Mais les opérations AWS KMS cryptographiques sont `KMSInvalidStateException` récurrentes pour toutes les défaillances liées à la configuration externe ou à l'état de connexion du magasin de clés externe. Pour identifier le problème, utilisez le texte du message d'erreur qui l'accompagne. L'ConnectCustomKeyStoreopération réussit rapidement avant que le processus de connexion ne soit terminé. Pour déterminer si le processus de connexion est réussi, consultez l'état de

connexion du magasin de clés externe. Si le processus de connexion échoue, AWS KMS renvoie un code d'erreur de connexion qui explique la cause et suggère une solution.

Rubriques

Outils de dépannage pour les magasins de clés externes

AWS KMS fournit plusieurs outils pour vous aider à identifier et à résoudre les problèmes liés à votre magasin de clés externe et à ses clés. Utilisez ces outils conjointement avec les outils fournis avec votre proxy de magasin de clés externe et votre gestionnaire de clés externe.

> **Note**
>
> Votre proxy de magasin de clés externe et votre gestionnaire de clés externe peuvent fournir des méthodes plus simples pour créer et gérer votre magasin de clés externe et ses clés KMS. Pour plus de détails, veuillez consulter la documentation de vos outils externes.

AWS KMS exceptions et messages d'erreur

AWS KMS fournit un message d'erreur détaillé concernant tout problème rencontré. Vous trouverez des informations supplémentaires sur les AWS KMS exceptions dans le manuel de référence des AWS Key Management Service API et AWS SDKs. Même si vous utilisez la AWS

KMS console, ces références peuvent vous être utiles. Par exemple, veuillez consulter la liste des erreurs correspondant à l'opération `CreateCustomKeyStores`.

Pour optimiser les performances de votre proxy de stockage de clés externe, AWS KMS renvoie les exceptions en fonction de la fiabilité de votre proxy au cours d'une période d'agrégation donnée de 5 minutes. En cas d'erreur interne du serveur 500, d'indisponibilité du service 503 ou d'expiration du délai de connexion, un proxy hautement fiable revient `KMSInternalException` et déclenche une nouvelle tentative automatique pour s'assurer que les demandes aboutissent finalement. Cependant, un proxy peu fiable donne des résultats`KMSInvalidStateException`. Pour plus d'informations, consultez la section Surveillance d'un magasin de clés externe.

Si le problème apparaît dans un autre AWS service, par exemple lorsque vous utilisez une clé KMS dans votre banque de clés externe pour protéger une ressource d'un autre AWS service, le AWS service peut fournir des informations supplémentaires pour vous aider à identifier le problème. Si le AWS service ne fournit pas le message, vous pouvez consulter le message d'erreur dans les CloudTrail journaux qui enregistrent l'utilisation de votre clé KMS.

CloudTrail journaux

Chaque opération AWS KMS d'API, y compris les actions dans la AWS KMS console, est enregistrée dans AWS CloudTrail des journaux. AWS KMS enregistre une entrée dans le journal des opérations réussies et échouées. Pour les opérations ayant échoué, l'entrée du journal inclut le nom de l'exception AWS KMS (`errorCode`) et le message d'erreur (`errorMessage`). Vous pouvez utiliser ces informations pour vous aider à identifier et résoudre l'erreur. Pour obtenir un exemple, consultez Échec lors du déchiffrement avec une clé KMS dans un magasin de clés externe.

L'entrée du journal inclut également l'ID de requête. Si la requête a atteint le proxy de votre magasin de clés externe, vous pouvez utiliser l'ID de requête indiqué dans l'entrée du journal pour rechercher la requête correspondante dans vos journaux de proxy, si votre proxy les fournit.

CloudWatch métriques

AWS KMS enregistre des CloudWatch statistiques Amazon détaillées concernant le fonctionnement et les performances de votre magasin de clés externe, notamment la latence, les limitations, les erreurs de proxy, le statut du gestionnaire de clés externe, le nombre de jours avant l'expiration de votre certificat TLS et l'âge indiqué de vos informations d'authentification de proxy. Vous pouvez utiliser ces mesures pour développer des modèles de données pour le fonctionnement de votre banque de clés externe et des CloudWatch alarmes qui vous alertent des problèmes imminents avant qu'ils ne surviennent.

> **Important**
>
> AWS KMS vous recommande de créer des CloudWatch alarmes pour surveiller les métriques du magasin de clés externe. Ces alertes vous signaleront les signes précurseurs de problèmes avant qu'ils ne se produisent.

Graphiques de surveillance

AWS KMS affiche des graphiques des CloudWatch statistiques du magasin de clés externe sur la page détaillée de chaque magasin de clés externe de la AWS KMS console. Vous pouvez utiliser les données des graphiques pour localiser la source des erreurs, détecter les problèmes imminents, établir des bases de référence et affiner vos seuils CloudWatch d'alarme. Pour plus de détails sur l'interprétation des graphiques de surveillance et l'utilisation de leurs données, veuillez consulter la rubrique Surveillez les magasins de clés externes.

Affichages des magasins de clés externes et des clés KMS

AWS KMS affiche des informations détaillées sur vos magasins de clés externes et les clés KMS dans le magasin de clés externe de la AWS KMS console, ainsi que dans la réponse aux DescribeKeyopérations DescribeCustomKeyStoreset. Ces affichages incluent des champs spéciaux pour les magasins de clés externes et les clés KMS contenant des informations que vous pouvez utiliser pour le dépannage, telles que l'état de connexion du magasin de clés externe et l'ID de la clé externe associée à la clé KMS. Pour en savoir plus, consultez Afficher les magasins de clés externes.

Client de test du proxy XKS (langue française non garantie)

AWS KMS fournit un client de test open source qui vérifie que votre proxy de stockage de clés externe est conforme à la spécification de l'API de proxy de stockage de clés AWS KMS externe. Vous pouvez utiliser ce client de test pour identifier et résoudre les problèmes liés au proxy de votre magasin de clés externe.

Erreurs de configuration

Lorsque vous créez un magasin de clés externe, vous spécifiez les valeurs des propriétés qui constituent la configuration de votre magasin de clés externe, telles que les informations d'identification pour l'authentification du proxy, le point de terminaison d'URI de proxy, le chemin d'URI de proxy et le nom du service de point de terminaison d'un VPC. Lorsqu'une erreur est AWS

KMS détectée dans la valeur d'une propriété, l'opération échoue et renvoie une erreur indiquant la valeur défectueuse.

De nombreux problèmes de configuration peuvent être résolus en corrigeant la valeur incorrecte. Vous pouvez corriger un chemin d'URI de proxy ou des informations d'identification pour l'authentification de proxy non valide sans déconnecter le magasin de clés externe. Pour les définitions de ces valeurs, y compris les exigences d'unicité, veuillez consulter la rubrique Rassembler les conditions requises. Pour obtenir des instructions sur la mise à jour de ces valeurs, veuillez consulter la rubrique Modifier les propriétés du magasin de clés externe.

Pour éviter les erreurs liées au chemin de l'URI de votre proxy et aux valeurs des informations d'identification pour l'authentification du proxy, lorsque vous créez ou mettez à jour votre magasin de clés externe, chargez un fichier de configuration du proxy sur la console AWS KMS . Il s'agit d'un fichier JSON, contenant le chemin d'URI de proxy et les valeurs d'informations d'identification pour l'authentification du proxy, fourni par le proxy de votre magasin de clés externe ou votre gestionnaire de clés externe. Vous ne pouvez pas utiliser un fichier de configuration de proxy avec des opérations d' AWS KMS API, mais vous pouvez utiliser les valeurs du fichier pour fournir des valeurs de paramètres pour vos demandes d'API qui correspondent aux valeurs de votre proxy.

Erreurs de configuration générale

Exceptions : `CustomKeyStoreInvalidStateException` (`CreateKey`), `KMSInvalidStateException` (opérations cryptographiques), `XksProxyInvalidConfigurationException` (opérations de gestion, à l'exception de `CreateKey`)

Codes d'erreur de connexion : `XKS_PROXY_INVALID_CONFIGURATION`, `XKS_PROXY_INVALID_TLS_CONFIGURATION`

Pour les magasins de clés externes connectés à un point de terminaison public, AWS KMS testez les valeurs des propriétés lorsque vous créez et mettez à jour le magasin de clés externe. Pour les magasins de clés externes dotés d'une connectivité au service de point de terminaison d'un VPC, AWS KMS teste les valeurs des propriétés lorsque vous connectez et mettez à jour le magasin de clés externe.

> **Note**
>
> L'opération `ConnectCustomKeyStore`, qui est asynchrone, peut réussir même si la tentative de connexion du magasin de clés externe à son proxy de magasin de clés externe échoue. Dans ce cas, il n'y a pas d'exception, mais l'état de connexion du magasin de clés

externe est Échec et un code d'erreur de connexion explique le message d'erreur. Pour de plus amples informations, veuillez consulter Erreurs de connexion au magasin de clés externe.

Si une erreur est AWS KMS détectée dans la valeur d'une propriété, l'opération échoue et renvoie `XksProxyInvalidConfigurationException` l'un des messages d'erreur suivants.

Le proxy de magasin de clés externe a rejeté la requête en raison d'un chemin d'URI non valide. Vérifiez le chemin de l'URI de votre magasin de clés externe et mettez-le à jour si nécessaire.

- Le chemin de l'URI du proxy est le chemin de base pour les AWS KMS demandes adressées au proxy APIs. Si ce chemin est incorrect, toutes les requêtes adressées au proxy échouent. Pour afficher le chemin actuel de l'URI de proxy pour votre magasin de clés externe, utilisez la console AWS KMS ou l'opération `DescribeCustomKeyStores`. Pour trouver le chemin d'URI de proxy correct, veuillez consulter la documentation relative au proxy de votre magasin de clés externe. Pour obtenir de l'aide pour corriger la valeur du chemin d'URI de votre proxy, veuillez consulter la rubrique Modifier les propriétés du magasin de clés externe.
- Le chemin d'URI de proxy pour le proxy de votre magasin de clés externe peut changer en fonction des mises à jour apportées à votre proxy de magasin de clés externe ou à votre gestionnaire de clés externe. Pour plus d'informations sur ces modifications, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre gestionnaire de clés externe.

`XKS_PROXY_INVALID_TLS_CONFIGURATION`
AWS KMS ne peut pas établir de connexion TLS avec le proxy de magasin de clés externe. Vérifiez la configuration TLS, y compris son certificat.

- Tous les proxys de magasin de clés externe nécessitent un certificat TLS. Le certificat TLS doit être émis par une autorité de certification publique (CA) prise en charge pour les magasins de clés externes. Pour la liste des autorités de certification prises en charge CAs, consultez la section Autorités de certification fiables dans la spécification de l'API proxy de stockage de clés AWS KMS externe.
- Pour la connectivité au point de terminaison public, le nom commun (CN) du sujet figurant sur le certificat TLS doit correspondre au nom de domaine indiqué dans le point de terminaison

d'URI de proxy pour le proxy de magasin de clés externe. Par exemple, si le point de terminaison public est https://myproxy.xks.example.com, le TLS, le CN du certificat TLS doit être `myproxy.xks.example.com` ou `*.xks.example.com`.

- Pour la connectivité au service de point de terminaison d'un VPC, le nom commun (CN) du sujet figurant sur le certificat TLS doit correspondre au nom DNS privé de votre service de point de terminaison d'un VPC. Par exemple, si le nom DNS privé est myproxy-private.xks.example.com, le CN du certificat TLS doit être `myproxy-private.xks.example.com` ou `*.xks.example.com`.
- Le certificat TLS ne peut pas avoir expiré. Pour obtenir la date d'expiration d'un certificat TLS, utilisez des outils SSL tels qu'OpenSSL. Pour surveiller la date d'expiration d'un certificat TLS associé à un magasin de clés externe, utilisez la XksProxyCertificateDaysToExpire CloudWatch métrique. Le nombre de jours avant la date d'expiration de votre certification TLS apparaît également dans la section Surveillance de la AWS KMS console.
- Si vous utilisez une connectivité au point de terminaison public, utilisez des outils de test SSL pour tester votre configuration SSL. Les erreurs de connexion TLS peuvent résulter d'un chaînage de certificats incorrect.

Erreurs de configuration de la connectivité au service de point de terminaison d'un VPC

Exceptions : `XksProxyVpcEndpointServiceNotFoundException`, `XksProxyVpcEndpointServiceInvalidConfigurationException`

Outre les problèmes de connectivité généraux, vous pouvez rencontrer les problèmes suivants lors de la création, de la connexion ou de la mise à jour d'un magasin de clés externe doté d'une connectivité au service de point de terminaison VPC. AWS KMS teste les valeurs des propriétés d'un magasin de clés externe avec connectivité au service de point de terminaison VPC lors de la création, de la connexion et de la mise à jour du magasin de clés externe. Lorsque les opérations de gestion échouent en raison d'erreurs de configuration, elles génèrent les exceptions suivantes :

```
XksProxyVpcEndpointServiceNotFoundException
```

Ce problème peut être dû à l'une des raisons suivantes :

- Nom de service de point de terminaison d'un VPC incorrect. Vérifiez que le nom du service de point de terminaison d'un VPC pour le magasin de clés externe est correct et qu'il correspond à la valeur de point de terminaison d'URI de proxy pour le magasin de clés externe. Pour trouver le nom du service de point de terminaison VPC, utilisez la console Amazon VPC ou l'opération.

DescribeVpcEndpointServices Pour trouver le nom du service de point de terminaison VPC et le point de terminaison URI du proxy d'un magasin de clés externe existant, utilisez la AWS KMS console ou l'DescribeCustomKeyStoresopération. Pour en savoir plus, consultez Afficher les magasins de clés externes.

- Le service de point de terminaison VPC peut se trouver dans un magasin de clés différent Région AWS de celui du magasin de clés externe. Vérifiez que le service de point de terminaison d'un VPC et le magasin de clés externe se trouvent dans la même région. (Le nom externe du nom de région, tel que, fait partie du nom du service de point de terminaison VPC`us-east-1`, tel que com.amazonaws.vpce.us-east-1. vpce-svc-example.) Pour obtenir la liste des exigences relatives au service de point de terminaison d'un VPC pour un magasin de clés externe, veuillez consulter la rubrique Service de point de terminaison d'un VPC. Vous ne pouvez pas déplacer un service de point de terminaison d'un VPC ou un magasin de clés externe vers une autre région. Vous pouvez toutefois créer un magasin de clés externe dans la même région que le service de point de terminaison d'un VPC. Pour plus d'informations, consultez Configurer la connectivité du service de point de terminaison VPC et Création d'un magasin de clés externe.
- AWS KMS n'est pas un principal autorisé pour le service de point de terminaison VPC. La liste Allow principals (Principaux autorisés) pour le service de point de terminaison d'un VPC doit inclure la valeur `cks.kms.<region>.amazonaws.com`, telle que `cks.kms.eu-west-3.amazonaws.com`. Pour obtenir des instructions sur l'ajout de cette valeur, veuillez consulter la rubrique Manage permissions (Gérer les autorisations) dans le Guide AWS PrivateLink .

XksProxyVpcEndpointServiceInvalidConfigurationException

Cette erreur se produit lorsque le service de point de terminaison d'un VPC ne répond pas à l'une des exigences suivantes :

- Le VPC nécessite au moins deux sous-réseaux privés, chacun dans une zone de disponibilité différente. Pour en savoir plus sur l'ajout d'un sous-réseau à votre VPC, veuillez consulter la rubrique Créer un sous-réseau dans votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Votre type de service de point de terminaison d'un VPC doit utiliser un équilibreur de charge réseau, et pas un équilibreur de charge de passerelle.
- L'acceptation ne doit pas être requise pour le service de point de terminaison d'un VPC (Acceptance required [Acceptation requise] ne doit pas être sélectionné). Si l'acceptation manuelle

de chaque demande de connexion est requise, vous AWS KMS ne pouvez pas utiliser le service de point de terminaison VPC pour vous connecter au proxy de banque de clés externe. Pour plus de détails, veuillez consulter la rubrique Accept or reject connection requests (Accepter ou rejeter les requêtes de connexion) dans le Guide AWS PrivateLink .

- Le service de point de terminaison d'un VPC doit avoir un nom DNS privé qui est un sous-domaine d'un domaine public. Par exemple, si le nom DNS privé est `https://myproxy-private.xks.example.com`, les domaines `xks.example.com` ou `example.com` doivent disposer d'un serveur DNS public. Pour afficher ou modifier le nom DNS privé de votre service de point de terminaison d'un VPC, veuillez consulter la rubrique Manage DNS names for VPC endpoint services (Gérer les noms DNS pour les services de point de terminaison d'un VPC) dans le Guide AWS PrivateLink .
- Le Domain verification status (Statut de vérification du domaine) du domaine de votre nom DNS privé doit être `verified`. Pour afficher et mettre à jour le statut de vérification du domaine de nom DNS privé, veuillez consulter la rubrique Étape 5 : Vérifiez votre nom de domaine DNS privé. Il peut s'écouler quelques minutes avant que le statut de vérification mis à jour n'apparaisse après que vous ayez ajouté l'enregistrement de texte requis.

  > **Note**
  >
  > Un domaine DNS privé ne peut être vérifié que s'il s'agit du sous-domaine d'un domaine public. Sinon, le statut de vérification du domaine DNS privé ne change pas, même après avoir ajouté l'enregistrement TXT requis.

- Assurez-vous que tous les pare-feux situés entre le proxy de stockage de clés externe AWS KMS et le proxy autorisent le trafic à destination et en provenance du port 443 du proxy. AWS KMS communique sur le port 443 IPv4. Cette valeur n'est pas configurable.
- Le nom DNS privé du service de point de terminaison d'un VPC doit correspondre à la valeur de point de terminaison d'URI de proxy pour le magasin de clés externe. Pour un magasin de clés externe doté d'une connectivité au service de point de terminaison d'un VPC, le point de terminaison d'URI de proxy doit être `https://` suivi du nom DNS privé du service de point de terminaison d'un VPC. Pour consulter la valeur du point de terminaison d'URI de proxy, veuillez consulter la rubrique Afficher les magasins de clés externes. Pour modifier la valeur du point de terminaison d'URI de proxy, veuillez consulter la rubrique Modifier les propriétés du magasin de clés externe.

Erreurs de connexion au magasin de clés externe

Le processus de connexion d'un magasin de clés externe à son proxy de magasin de clés externe prend environ cinq minutes. Sauf si elle échoue rapidement, l'opération `ConnectCustomKeyStore` renvoie une réponse HTTP 200 et un objet JSON sans propriétés. Cependant, cette réponse initiale n'indique pas que la connexion a abouti. Pour déterminer l'état de connexion du magasin de clés externe, référez-vous à son état de connexion. Si la connexion échoue, l'état de connexion de la banque de clés externe AWS KMS devient `FAILED` et renvoie un code d'erreur de connexion expliquant la cause de l'échec.

> **Note**
>
> Si l'état de connexion d'un magasin de clés personnalisé est `FAILED`, vous devez déconnecter le magasin de clés personnalisé avant de le reconnecter. Vous ne pouvez pas connecter un magasin de clés personnalisé avec un statut de connexion `FAILED`.

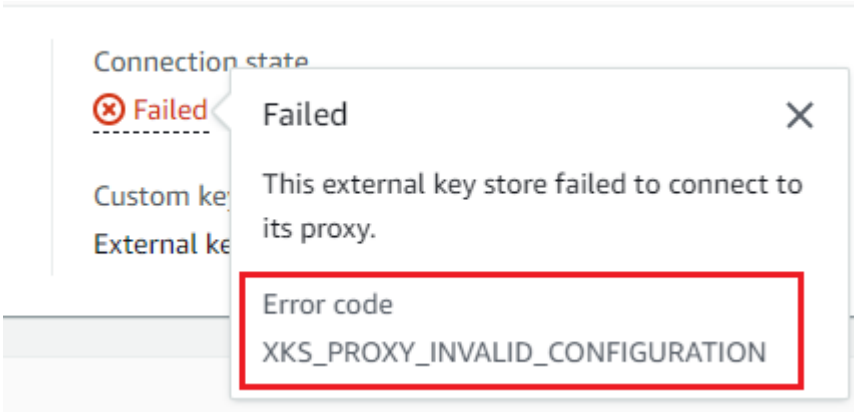
Pour consulter l'état de connexion d'un magasin de clés externe :

- Dans la DescribeCustomKeyStoresréponse, visualisez la valeur de l'`ConnectionState`élément.
- Dans la AWS KMS console, l'état de connexion apparaît dans le tableau du magasin de clés externe. De plus, sur la page détaillée de chaque magasin de clés externe, Connection state (État de la connexion) apparaît dans la section General configuration (Configuration générale).

Lorsque l'état de connexion est `FAILED`, le code d'erreur de connexion permet d'expliquer l'erreur.

Pour consulter le code d'erreur de connexion, procédez comme suit :

- Dans la DescribeCustomKeyStoresréponse, visualisez la valeur de l'`ConnectionErrorCode`élément. Cet élément apparaît dans la réponse de `DescribeCustomKeyStores` uniquement lorsque le `ConnectionState` est `FAILED`.
- Pour afficher le code d'erreur de connexion dans la AWS KMS console, sur la page détaillée du magasin de clés externe, passez le curseur sur la valeur Échec.



Codes d'erreur de connexion pour les magasins de clés externes

Les codes d'erreur de connexion suivants s'appliquent aux magasins de clés externes.

`INTERNAL_ERROR`

AWS KMS Impossible de terminer la demande en raison d'une erreur interne. Réitérez la demande. Pour les demandes `ConnectCustomKeyStore`, déconnectez le magasin de clés personnalisé avant de tenter de vous connecter à nouveau.

`INVALID_CREDENTIALS`

L'une ou les deux valeurs de `XksProxyAuthenticationCredential` ne sont pas valides sur le proxy de magasin de clés externe spécifié.

`NETWORK_ERRORS`

Des erreurs réseau AWS KMS empêchent de connecter le magasin de clés personnalisé à son magasin de clés secondaire.

`XKS_PROXY_ACCESS_DENIED`

AWS KMS les demandes se voient refuser l'accès au proxy de stockage de clés externe. Si le proxy de magasin de clés externe possède des règles d'autorisation, vérifiez qu'elles autorisent AWS KMS à communiquer avec le proxy en votre nom.

`XKS_PROXY_INVALID_CONFIGURATION`

Une erreur de configuration empêche le magasin de clés externe de se connecter à son proxy. Vérifiez la valeur du `XksProxyUriPath`.

`XKS_PROXY_INVALID_RESPONSE`

AWS KMS Impossible d'interpréter la réponse du proxy de stockage de clés externe. Si ce code d'erreur de connexion s'affiche à plusieurs reprises, informez-en le fournisseur du proxy de votre magasin de clés externe.

`XKS_PROXY_INVALID_TLS_CONFIGURATION`

AWS KMS Impossible de se connecter au proxy de banque de clés externe car la configuration TLS n'est pas valide. Vérifiez que le proxy de magasin de clés externe prend en charge le protocole TLS 1.2 ou 1.3. Vérifiez également que le certificat TLS n'a pas expiré, qu'il correspond au nom d'hôte indiqué dans la valeur de `XksProxyUriEndpoint` et qu'il est signé par une autorité de certification approuvée figurant dans la liste des Autorités de certification approuvées (langue française non garantie).

`XKS_PROXY_NOT_REACHABLE`

AWS KMS ne peut pas communiquer avec votre proxy de stockage de clés externe. Vérifiez que les `XksProxyUriEndpoint` et `XksProxyUriPath` sont corrects. Utilisez les outils de votre proxy de magasin de clés externe pour vérifier que le proxy est actif et disponible sur son réseau. Vérifiez également que vos instances de gestionnaire de clés externe fonctionnent correctement. Les tentatives de connexion échouent avec ce code d'erreur de connexion si le proxy indique qu'aucune instance du gestionnaire de clés externe n'est disponible.

`XKS_PROXY_TIMED_OUT`

AWS KMS peut se connecter au proxy de stockage de clés externe, mais le proxy ne répond pas AWS KMS dans le délai imparti. Si ce code d'erreur de connexion s'affiche à plusieurs reprises, informez-en le fournisseur du proxy de votre magasin de clés externe.

`XKS_VPC_ENDPOINT_SERVICE_INVALID_CONFIGURATION`

La configuration du service de point de terminaison Amazon VPC n'est pas conforme aux exigences d'un magasin de clés AWS KMS externe.

- Le service de point de terminaison d'un VPC doit être un service de point de terminaison pour les points de terminaison d'interface dans l' Compte AWS de l'appelant.
- Il doit comporter un équilibreur de charge réseau (NLB) connecté à au moins deux sous-réseaux, dans deux zones de disponibilités distinctes.
- La `Allow principals` liste doit inclure le principal de AWS KMS service de la régioncks.kms.<region>.amazonaws.com, tel quecks.kms.us-east-1.amazonaws.com.

- L'acceptation des requêtes de connexion ne doit pas être requise.
- Il doit avoir un nom DNS privé. Le nom DNS privé d'un magasin de clés externe avec une connectivité `VPC_ENDPOINT_SERVICE` doit être unique dans sa Région AWS.
- La valeur de statut de vérification du domaine du nom DNS privé doit être `verified`.
- Le certificat TLS spécifie le nom d'hôte DNS privé auquel le point de terminaison est accessible.

`XKS_VPC_ENDPOINT_SERVICE_NOT_FOUND`

AWS KMS ne trouve pas le service de point de terminaison VPC qu'il utilise pour communiquer avec le proxy de banque de clés externe. Vérifiez que le `XksProxyVpcEndpointServiceName` est correct et que le principal du service AWS KMS dispose des autorisations de consommateur de service sur le service de point de terminaison d'un Amazon VPC.

Erreurs de latence et de délai d'expiration

Exceptions : `CustomKeyStoreInvalidStateException` (`CreateKey`), `KMSInvalidStateException` (opérations cryptographiques), `XksProxyUriUnreachableException` (opérations de gestion)

Codes d'erreur de connexion : `XKS_PROXY_NOT_REACHABLE`, `XKS_PROXY_TIMED_OUT`

Lorsque vous AWS KMS ne parvenez pas à contacter le proxy dans le délai d'expiration de 250 millisecondes, il renvoie une exception. `CreateCustomKeyStore`et `UpdateCustomKeyStore` revenez`XksProxyUriUnreachableException`. Les opérations cryptographiques renvoient la norme `KMSInvalidStateException` avec un message d'erreur décrivant le problème. En cas d'`ConnectCustomKeyStore`échec, AWS KMS renvoie un code d'erreur de connexion décrivant le problème.

Les erreurs de délai d'expiration peuvent être des problèmes transitoires pouvant être résolus en réitérant la requête. Si le problème persiste, vérifiez que le proxy de votre magasin de clés externe est actif et connecté au réseau, et que le point de terminaison d'URI de proxy, le chemin d'URI de proxy et le nom du service de point de terminaison d'un VPC (le cas échéant) sont corrects dans votre magasin de clés externe. Vérifiez également que votre gestionnaire de clés externe est proche de celui Région AWS de votre magasin de clés externe. Si vous devez mettre à jour l'une de ces valeurs, veuillez consulter la rubrique Modifier les propriétés du magasin de clés externe.

Pour suivre les modèles de latence, utilisez la `XksProxyLatency` CloudWatch métrique et le graphique de latence moyenne (basé sur cette métrique) dans la section Surveillance de la AWS

KMS console. Votre proxy de magasin de clés externe peut également générer des journaux et des métriques permettant de suivre la latence et les délais d'expiration.

`XksProxyUriUnreachableException`
AWS KMS ne peut pas communiquer avec le proxy de stockage de clés externe. Il peut s'agir d'un problème réseau transitoire. Si cette erreur s'affiche à plusieurs reprises, vérifiez que le proxy de votre magasin de clés externe est actif et connecté au réseau, et que l'URI de son point de terminaison est correcte dans votre magasin de clés externe.

- Le proxy de stockage de clés externe n'a pas répondu à une demande d'API AWS KMS proxy dans le délai de 250 millisecondes. Cela peut indiquer un problème réseau transitoire ou un problème de fonctionnement ou de performance avec le proxy. Si une nouvelle tentative ne résout pas le problème, prévenez l'administrateur de votre proxy de magasin de clés externe.

Les erreurs de latence et de délai d'expiration se manifestent souvent par des échecs de connexion. Lorsque l'ConnectCustomKeyStoreopération échoue, l'état de connexion du magasin de clés externe AWS KMS devient `FAILED` et renvoie un code d'erreur de connexion expliquant l'erreur. Pour obtenir la liste des codes d'erreur de connexion et des suggestions pour les résoudre, veuillez consulter la rubrique Codes d'erreur de connexion pour les magasins de clés externes. Les listes de codes de connexion pour All custom key stores (Tous les magasins de clés personnalisés) et les External key stores (Magasins de clés externes) s'appliquent aux magasins de clés externes. Les erreurs de connexion suivantes sont liées à la latence et aux délais d'expiration.

`XKS_PROXY_NOT_REACHABLE`

-ou-

`CustomKeyStoreInvalidStateException` , `KMSInvalidStateException` , `XksProxyUriUnreachableException`
AWS KMS ne peut pas communiquer avec le proxy de stockage de clés externe. Vérifiez que le proxy de votre magasin de clés externe est actif et connecté au réseau, et que son chemin d'URI et son URI de point de terminaison ou son nom de service VPC sont corrects dans votre magasin de clés externe.

Cette erreur peut se produire dans les conditions suivantes :

- Le proxy de magasin de clés externe n'est pas actif et/ou n'est pas connecté au réseau.
- Une erreur s'est produite dans les valeurs du point de terminaison d'URI de proxy, du chemin d'URI de proxy ou du nom du service de point de terminaison d'un VPC (le cas échéant) dans la configuration du magasin de clés externe. Pour consulter la configuration du magasin de clés externe, utilisez l'DescribeCustomKeyStoresopération ou consultez la page détaillée du magasin de clés externe dans la AWS KMS console.
- Il se peut qu'une erreur de configuration réseau, telle qu'une erreur de port, se produise sur le chemin réseau entre le proxy de stockage de clés externe AWS KMS et le proxy de stockage de clés. AWS KMS communique avec le proxy de stockage de clés externe sur le port 443 IPv4. Cette valeur n'est pas configurable.
- Lorsque le proxy de stockage de clés externe indique (dans une GetHealthStatusréponse) que toutes les instances du gestionnaire de clés externe le sontUNAVAILABLE, l'ConnectCustomKeyStoreopération échoue avec un `ConnectionErrorCode` de`XKS_PROXY_NOT_REACHABLE`. Pour obtenir de l'aide, veuillez consulter la documentation de votre gestionnaire de clés externe.
- Cette erreur peut être due à une longue distance physique entre le gestionnaire de clés externe et le Région AWS magasin de clés externe. La latence du ping (temps d'aller-retour du réseau (RTT)) entre le gestionnaire de clés Région AWS et le gestionnaire de clés externe ne doit pas dépasser 35 millisecondes. Vous devrez peut-être créer un magasin de clés externe dans un Région AWS magasin plus proche du gestionnaire de clés externe, ou déplacer le gestionnaire de clés externe vers un centre de données plus proche du Région AWS.

`XKS_PROXY_TIMED_OUT`

-ou-

`CustomKeyStoreInvalidStateException` , `KMSInvalidStateException` , `XksProxyUriUnreachableException`
AWS KMS a rejeté la requête, car le proxy de magasin de clés externe n'a pas répondu dans les temps. Réitérez la demande. Si cette erreur s'affiche à plusieurs reprises, signalez-la à l'adminis trateur de votre proxy de magasin de clés externe.

Cette erreur peut se produire dans les conditions suivantes :

- Cette erreur peut résulter d'une longue distance physique entre le gestionnaire de clés externe et le proxy du magasin de clés externe. Si possible, rapprochez le proxy du magasin de clés externe du gestionnaire de clés externe.
- Des erreurs de temporisation peuvent survenir lorsque le proxy n'est pas conçu pour gérer le volume et la fréquence des demandes provenant de AWS KMS. Si vos CloudWatch statistiques indiquent un problème persistant, informez-en l'administrateur proxy de votre magasin de clés externe.
- Des erreurs de délai d'expiration peuvent survenir lorsque la connexion entre le gestionnaire de clés externe et l'Amazon VPC pour le magasin de clés externe ne fonctionne pas correctement. Si vous en utilisez AWS Direct Connect, vérifiez que votre VPC et votre gestionnaire de clés externe peuvent communiquer efficacement. Pour obtenir de l'aide pour résoudre les problèmes, consultez la section Résolution des problèmes AWS Direct Connect dans le guide de AWS Direct Connect l'utilisateur.

`XKS_PROXY_TIMED_OUT`

-ou-

`CustomKeyStoreInvalidStateException` , `KMSInvalidStateException` , `XksProxyUriUnreachableException`
Le proxy de magasin de clés externe n'a pas répondu à la requête dans le délai imparti. Réitérez la demande. Si cette erreur s'affiche à plusieurs reprises, signalez-la à l'administrateur de votre proxy de magasin de clés externe.

- Cette erreur peut résulter d'une longue distance physique entre le gestionnaire de clés externe et le proxy du magasin de clés externe. Si possible, rapprochez le proxy du magasin de clés externe du gestionnaire de clés externe.

Erreurs liées aux informations d'identification pour l'authentification

Exceptions : `CustomKeyStoreInvalidStateException` (`CreateKey`), `KMSInvalidStateException` (opérations cryptographiques), `XksProxyIncorrectAuthenticationCredentialException` (opérations de gestion autres que `CreateKey`)

Vous établissez et maintenez un identifiant d'authentification pour AWS KMS votre proxy de magasin de clés externe. Vous indiquez ensuite AWS KMS les valeurs d'identification lorsque vous créez un magasin de clés externe. Pour modifier les informations d'identification pour l'authentification, effectuez la modification sur votre proxy de magasin de clés externe. Ensuite, mettez à jour les informations d'identification de votre magasin de clés externe. Si votre proxy effectue une rotation des informations d'identification, vous devez mettre à jour les informations d'identification de votre magasin de clés externe.

Si le proxy de magasin de clés externe n'authentifie pas une requête signée avec les informations d'identification pour l'authentification du proxy de votre magasin de clés externe, le résultat dépend de la requête :

- `CreateCustomKeyStore` et `UpdateCustomKeyStore` échouent avec une exception `XksProxyIncorrectAuthenticationCredentialException`.
- `ConnectCustomKeyStore` réussit, mais la connexion échoue. L'état de connexion est `FAILED` et le code d'erreur de connexion est `INVALID_CREDENTIALS`. Pour en savoir plus, consultez Erreurs de connexion au magasin de clés externe.
- Les opérations cryptographiques renvoient toutes `KMSInvalidStateException` les erreurs de configuration externes et les erreurs d'état de connexion dans un magasin de clés externe. Le message d'erreur qui l'accompagne décrit le problème.

Le proxy de magasin de clés externe a rejeté la requête, car il n'a pas pu authentifier AWS KMS. Vérifiez les informations d'identification de votre magasin de clés externe et mettez-les à jour si nécessaire.

Cette erreur peut se produire dans les conditions suivantes :

- L'ID de clé d'accès ou la clé d'accès secrète du magasin de clés externe ne correspond pas aux valeurs établies sur le proxy de magasin de clés externe.

  Pour corriger cette erreur, mettez à jour les informations d'identification pour l'authentification du proxy de votre magasin de clés externe. Vous pouvez effectuer cette modification sans déconnecter votre magasin de clés externe.
- Un proxy inverse entre le proxy de stockage de clés externe AWS KMS et le proxy externe peut manipuler les en-têtes HTTP d'une manière qui invalide les signatures SigV4. Pour corriger cette erreur, contactez l'administrateur de votre proxy.

Erreurs d'état des clés

Exceptions : `KMSInvalidStateException`

`KMSInvalidStateException` est utilisée à deux fins distinctes pour les clés KMS dans les magasins de clés personnalisés.

- Lorsqu'une opération de gestion, telle que `CancelKeyDeletion`, échoue et renvoie cette exception, cela indique que l'état de clé de la clé KMS n'est pas compatible avec l'opération.
- Lorsqu'une opération cryptographique sur une clé KMS dans un magasin de clés personnalisé échoue avec l'exception `KMSInvalidStateException`, cela peut indiquer un problème lié à l'état de la clé KMS. Mais les opérations AWS KMS cryptographiques renvoient `KMSInvalidStateException` à toutes les erreurs de configuration externes et aux erreurs d'état de connexion dans un magasin de clés externe. Pour identifier le problème, utilisez le message d'erreur qui accompagne l'exception.

Pour trouver l'état clé requis pour les opérations d'une AWS KMS API, consultezÉtats clés des AWS KMS clés. Pour obtenir l'état de clé d'une clé KMS, sur la page Clés gérées par le client, veuillez consulter le champ Status (État) de la clé KMS. Vous pouvez également utiliser l'DescribeKeyopération et afficher l'`KeyState`élément dans la réponse. Pour en savoir plus, consultez Identifier et afficher les clés.

> **Note**
>
> L'état d'une clé KMS dans un magasin de clés externe n'indique rien quant au statut de la clé externe associée. Pour plus d'informations sur le statut de la clé externe, utilisez votre gestionnaire de clés externe et les outils de proxy de votre magasin de clés externe. L'exception `CustomKeyStoreInvalidStateException` fait référence à l'état de connexion du magasin de clés externe, et non à l'état de clé d'une clé KMS.

Une opération de chiffrement sur une clé KMS figurant dans un magasin personnalisé peut échouer si la clé KMS affiche l'état `Unavailable` ou `PendingDeletion`. (Les touches désactivées renvoient l'exception `DisabledException`).

- Une clé KMS possède un état `Disabled` clé uniquement lorsque vous la désactivez intentionnellement dans la AWS KMS console ou en utilisant l'DisableKeyopération. Lorsqu'une clé KMS est désactivée, vous pouvez afficher et gérer la clé, mais vous ne pouvez pas l'utiliser

dans les opérations cryptographiques. Pour résoudre ce problème, activez la clé. Pour en savoir plus, consultez Activer et désactiver les touches.

- L'état de clé d'une clé KMS est `Unavailable` lorsque le magasin de clés externe est déconnecté de son proxy de magasin de clés externe. Pour corriger une clé KMS indisponible, reconnectez le magasin de clés externe. Une fois le magasin de clés externe reconnecté, l'état de clé des clés KMS du magasin de clés externe est automatiquement restauré à son état précédent, soit `Enabled` ou `Disabled`.

  L'état de clé d'une clé KMS est `PendingDeletion` lorsque sa suppression a été planifiée et qu'elle se trouve dans sa période d'attente. Une erreur d'état de clé sur une clé KMS en attente de suppression indique que la clé ne doit pas être supprimée, soit parce qu'elle est utilisée pour le chiffrement, soit parce qu'elle est requise pour le déchiffrement. Pour réactiver la clé KMS, annulez la suppression planifiée, puis activez la clé. Pour en savoir plus, consultez Planifier la suppression des clés.

Erreurs de déchiffrement

Exceptions : `KMSInvalidStateException`

Lorsqu'une opération de déchiffrement avec une clé KMS dans un magasin de clés externe échoue, AWS KMS renvoie la norme utilisée par `KMSInvalidStateException` les opérations cryptographiques pour toutes les erreurs de configuration externes et les erreurs d'état de connexion sur un magasin de clés externe. Le message d'erreur signale le problème.

Pour déchiffrer un texte chiffré à l'aide d'un double chiffrement, le gestionnaire de clés externes utilise d'abord la clé externe pour déchiffrer la couche externe du texte chiffré. AWS KMS Utilise ensuite le contenu AWS KMS clé de la clé KMS pour déchiffrer la couche interne de texte chiffré. Un texte chiffré non valide ou endommagé peut être rejeté par le gestionnaire de clés externe ou par AWS KMS.

Les messages d'erreur suivants accompagnent l'exception `KMSInvalidStateException` en cas d'échec du déchiffrement. Cela indique un problème lié au texte chiffré ou au contexte de chiffrement facultatif de la requête.

> Le proxy de magasin de clés externe a rejeté la requête, car le texte chiffré spécifié ou les données authentifiées supplémentaires sont endommagés, manquants ou non valides.

- Lorsque le proxy de stockage de clés externe ou le gestionnaire de clés externe signalent qu'un texte chiffré ou son contexte de chiffrement n'est pas valide, cela indique généralement un problème lié au texte chiffré ou au contexte de chiffrement dans la `Decrypt` demande envoyée à. AWS KMS Pour les `Decrypt` opérations, AWS KMS envoie au proxy le même texte chiffré et le même contexte de chiffrement qu'il reçoit dans la `Decrypt` demande.

  Cette erreur peut être causée par un problème de mise en réseau lors du transit, comme une inversion de bit. Réitérez la requête `Decrypt`. Si le problème persiste, vérifiez que le texte chiffré n'a pas été altéré ou corrompu. Vérifiez également que le contexte de chiffrement de la `Decrypt` demande AWS KMS correspond au contexte de chiffrement de la demande qui a chiffré les données.

Le texte chiffré que le proxy de magasin de clés externe a soumis pour déchiffrement, ou le contexte de chiffrement, est endommagé, manquant ou non valide.

- Lorsqu'il AWS KMS rejette le texte chiffré reçu du proxy, cela indique que le gestionnaire de clés externe ou le proxy a renvoyé un texte chiffré non valide ou endommagé à. AWS KMS

  Cette erreur peut être causée par un problème de mise en réseau lors du transit, comme une inversion de bit. Réitérez la requête `Decrypt`. Si le problème persiste, vérifiez que le gestionnaire de clés externe fonctionne correctement et que le proxy de stockage de clés externe ne modifie pas le texte chiffré qu'il reçoit du gestionnaire de clés externe avant de le renvoyer. AWS KMS

Erreurs liées aux clés externes

Une clé externe est une clé cryptographique du gestionnaire de clés externe qui fait office d'éléments de clé externes pour une clé KMS. AWS KMS ne peut pas accéder directement à la clé externe. Il doit demander au gestionnaire de clés externe (via le proxy de magasin de clés externe) d'utiliser la clé externe pour chiffrer des données ou déchiffrer un texte chiffré.

Vous spécifiez l'ID de la clé externe dans son gestionnaire de clés externe lorsque vous créez une clé KMS dans votre magasin de clés externe. Vous ne pouvez pas modifier l'ID de la clé externe après la création de la clé KMS. Pour éviter tout problème lié à la clé KMS, l'opération `CreateKey` demande au proxy de magasin de clés externe de vérifier l'ID et la configuration de la clé externe. Si la clé externe ne répond pas aux exigences requises pour être utilisée avec une clé KMS, l'opération `CreateKey` échoue avec une exception et un message d'erreur identifiant le problème.

Cependant, des problèmes peuvent survenir après la création de la clé KMS. Si une opération de chiffrement échoue en raison d'un problème lié à la clé externe, elle renvoie une exception `KMSInvalidStateException` avec un message d'erreur indiquant le problème.

CreateKey erreurs pour la clé externe

Exceptions : `XksKeyAlreadyInUseException`, `XksKeyNotFoundException`, `XksKeyInvalidConfigurationException`

L'CreateKeyopération tente de vérifier l'ID et les propriétés de la clé externe que vous fournissez dans le paramètre ID de clé externe (console) ou `XksKeyId` (API). Cette pratique a pour but de détecter les erreurs à un stade précoce avant que vous n'essayiez d'utiliser la clé externe avec la clé KMS.

Clé externe en cours d'utilisation

Chaque clé KMS d'un magasin de clés externe doit utiliser une clé externe différente. Lorsqu'il `CreateKey` reconnaît que l'ID de clé externe (XksKeyId) d'une clé KMS n'est pas unique dans le magasin de clés externe, il échoue avec un`XksKeyAlreadyInUseException`.

Si vous en utilisez plusieurs IDs pour la même clé externe, vous `CreateKey` ne reconnaîtrez pas le doublon. Cependant, les clés KMS associées à la même clé externe ne sont pas interopérables car elles contiennent des AWS KMS éléments clés et des métadonnées différents.

Clé externe introuvable

Lorsque le proxy de stockage de clés externe indique qu'il ne peut pas trouver la clé externe à l'aide de l'ID de clé externe (XksKeyId) pour la clé KMS, l'`CreateKey`opération échoue et renvoie `XksKeyNotFoundException` le message d'erreur suivant.

> Le proxy de magasin de clés externe a rejeté la requête, car il n'a pas trouvé la clé externe.

Cette erreur peut se produire dans les conditions suivantes :

- L'ID de la clé externe (`XksKeyId`) de la clé KMS n'est peut-être pas valide. Pour trouver l'ID que votre proxy de clé externe utilise pour identifier la clé externe, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre gestionnaire de clés externe.
- La clé externe peut avoir été supprimée de votre gestionnaire de clés externe. Pour effectuer des recherches, utilisez vos outils de gestionnaire de clés externe. Si la clé externe est supprimée

définitivement, utilisez une autre clé externe avec la clé KMS. Pour obtenir la liste des exigences relatives à la clé externe, veuillez consulter la rubrique Exigences relatives à une clé KMS dans un magasin de clés externe.

Exigences relatives aux clés externes non satisfaites

Lorsque le proxy du magasin de clés externes indique que la clé externe ne répond pas aux exigences définies pour une utilisation avec une clé KMS, l'opération `CreateKey` échoue et renvoie l'exception `XksKeyInvalidConfigurationException` avec l'un des messages d'erreur suivants.

La spécification de clé de la clé externe doit être AES_256. La spécification clé de la clé externe spécifiée est*<key-spec>* .

- La clé externe doit être une clé de chiffrement symétrique de 256 bits avec une spécification de clé AES_256. Si la clé externe spécifiée est d'un type différent, spécifiez l'ID d'une clé externe qui répond à cette exigence.

Le statut de la clé externe doit être ACTIVÉ. L'état de la clé externe spécifiée est*<status>*.

- La clé externe doit être activée dans le gestionnaire de clés externe. Si la clé externe spécifiée n'est pas activée, utilisez vos outils de gestionnaire de clés externe pour l'activer ou spécifiez une clé externe activée.

L'utilisation de la clé externe doit inclure ENCRYPT et DECRYPT. La clé d'utilisation de la clé externe spécifiée est < *key-usage* >.

- La clé externe doit être configurée pour le chiffrement et le déchiffrement dans le gestionnaire de clés externe. Si la clé externe spécifiée n'inclut pas ces opérations, utilisez vos outils de gestionnaire de clés externe pour modifier les opérations ou spécifiez une autre clé externe.

Erreurs d'opérations cryptographiques pour la clé externe

Exceptions : `KMSInvalidStateException`

Lorsque le proxy de magasin de clés externe ne trouve pas la clé externe associée à la clé KMS, ou que la clé externe ne répond pas aux exigences requises pour être utilisée avec une clé KMS, l'opération cryptographique échoue.

Les problèmes de clé externe détectés lors d'une opération cryptographique sont plus difficiles à résoudre que les problèmes de clé externe détectés avant la création de la clé KMS. Vous ne pouvez pas modifier l'ID de la clé externe après la création de la clé KMS. Si la clé KMS n'a encore chiffré aucune donnée, vous pouvez supprimer la clé KMS et en créer une nouvelle avec un ID de clé externe différent. Cependant, le texte chiffré généré à l'aide de la clé KMS ne peut être déchiffré par aucune autre clé KMS, même avec la même clé externe, car les clés auront des métadonnées et des éléments clés différents. AWS KMS Dans la mesure du possible, utilisez plutôt vos outils de gestionnaire de clés externe pour résoudre le problème lié à la clé externe.

Lorsque le proxy du magasin de clés externes signale un problème lié à la clé externe, les opérations de chiffrement renvoient l'exception `KMSInvalidStateException` avec un message d'erreur identifiant le problème.

Clé externe introuvable

Lorsque le proxy de stockage de clés externe indique qu'il ne peut pas trouver la clé externe à l'aide de l'ID de clé externe (XksKeyId) de la clé KMS, les opérations cryptographiques renvoient un `KMSInvalidStateException` avec le message d'erreur suivant.

```
Le proxy de magasin de clés externe a rejeté la requête, car il n'a pas trouvé la clé externe.
```

Cette erreur peut se produire dans les conditions suivantes :

- L'ID de la clé externe (`XksKeyId`) de la clé KMS n'est plus valide.

  Pour trouver l'ID de clé externe associé à votre clé KMS, consultez les détails de la clé KMS. Pour trouver l'ID que votre proxy de clé externe utilise pour identifier la clé externe, veuillez consulter la documentation de votre proxy de magasin de clés externe ou de votre gestionnaire de clés externe.

  AWS KMS vérifie l'ID de clé externe lorsqu'il crée une clé KMS dans un magasin de clés externe. Cependant, l'ID peut devenir invalide, en particulier si la valeur de l'ID de clé externe est un alias ou un nom mutable. Vous ne pouvez pas modifier l'ID de clé externe associé à une clé KMS existante. Pour déchiffrer tout texte chiffré au moyen de la clé KMS, vous devez réassocier la clé externe à l'ID de la clé externe existante.

Si vous n'avez pas encore utilisé la clé KMS pour chiffrer des données, vous pouvez créer une clé KMS avec un ID de clé externe valide. Toutefois, si vous avez généré du texte chiffré à l'aide de la clé KMS, vous ne pouvez utiliser aucune autre clé KMS pour le déchiffrer, même si vous utilisez la même clé externe.

- La clé externe peut avoir été supprimée de votre gestionnaire de clés externe. Pour effectuer des recherches, utilisez vos outils de gestionnaire de clés externe. Si possible, essayez de récupérer les éléments de clé à partir d'une copie ou d'une sauvegarde de votre gestionnaire de clés externe. Si la clé externe est supprimée définitivement, tout texte chiffré au moyen de la clé KMS associée devient irrécupérable.

Erreurs de configuration des clés externes

Lorsque le proxy du magasin de clés externes indique que la clé externe ne répond pas aux exigences définies pour une utilisation avec une clé KMS, l'opération de chiffrement renvoie l'exception `KMSInvalidStateException` avec l'un des messages d'erreur suivants.

Le proxy de magasin de clés externe a rejeté la requête, car la clé externe ne prend pas en charge l'opération demandée.

- La clé externe doit prendre en charge à la fois le chiffrement et le déchiffrement. Si l'utilisation de la clé n'inclut pas le chiffrement et le déchiffrement, utilisez vos outils de gestionnaire de clés externe pour modifier l'utilisation de la clé.

Le proxy de magasin de clés externe a rejeté la requête, car la clé externe n'est pas activée dans le gestionnaire de clés externe.

- La clé externe doit être activée et disponible pour son utilisation dans le gestionnaire de clés externe. Si l'état de la clé externe n'est pas `Enabled`, utilisez vos outils de gestionnaire de clés externe pour l'activer.

Problèmes liés aux proxys

Exceptions :

`CustomKeyStoreInvalidStateException` (`CreateKey`), `KMSInvalidStateException` (opérations cryptographiques), `UnsupportedOperationException`, `XksProxyUriUnreachableException`, `XksProxyInvalidResponseException` (opérations de gestion autres que `CreateKey`)

Le proxy de stockage de clés externe assure la médiation de toutes les communications entre AWS KMS et le gestionnaire de clés externe. Il traduit les AWS KMS demandes génériques dans un format compréhensible par votre gestionnaire de clés externe. Si le proxy de stockage de clés externe n'est pas conforme à la spécification de l'API du proxy de stockage de clés AWS KMS externe, s'il ne fonctionne pas correctement ou s'il ne peut pas communiquer avec lui AWS KMS, vous ne pourrez pas créer ou utiliser de clés KMS dans votre magasin de clés externe.

Bien que de nombreuses erreurs mentionnent le proxy de magasin de clés externe en raison de son rôle critique dans l'architecture du magasin de clés externe, ces problèmes peuvent provenir du gestionnaire de clés externe ou de la clé externe.

Les problèmes abordés dans cette section concernent des problèmes liés à la conception ou au fonctionnement du proxy de magasin de clés externe. La résolution de ces problèmes peut nécessiter une modification du logiciel de proxy. Consultez l'administrateur de votre proxy. Pour vous aider à diagnostiquer les problèmes de proxy, AWS KMS fournit XKS Proxy Text Client, un client de test open source qui vérifie si votre proxy de magasin de clés externe respecte la spécification de l'API du proxy de magasin de clés externe AWS KMS (langue française non garantie).

`CustomKeyStoreInvalidStateException`, `KMSInvalidStateException` ou `XksProxyUriUnreachableException`
Le proxy de magasin de clés externe est dans un état défectueux. Si ce message s'affiche à plusieurs reprises, prévenez l'administrateur de votre proxy de magasin de clés externe.

- Cette erreur peut indiquer un problème de fonctionnement ou une erreur logicielle dans le proxy de magasin de clés externe. Vous pouvez trouver les entrées du CloudTrail journal correspondant à l'opération d' AWS KMS API qui a généré chaque erreur. Cette erreur peut être résolue en réitérant l'opération. Toutefois, si le problème persiste, prévenez l'administrateur de votre proxy de magasin de clés externe.
- Lorsque le proxy de banque de clés externe indique (dans une GetHealthStatusréponse) que toutes les instances du gestionnaire de clés externe le sont`UNAVAILABLE`, les tentatives de création ou de mise à jour d'une banque de clés externe échouent, à cette exception près. Si cette erreur persiste, consultez la documentation de votre gestionnaire de clés externe.

`CustomKeyStoreInvalidStateException` , `KMSInvalidStateException` ou `XksProxyInvalidResponseException`
AWS KMS Impossible d'interpréter la réponse du proxy de stockage de clés externe. Si cette erreur s'affiche à plusieurs reprises, contactez l'administrateur de votre proxy de magasin de clés externe.

- AWS KMS les opérations génèrent cette exception lorsque le proxy renvoie une réponse non définie qui AWS KMS ne peut ni analyser ni interpréter. Cette erreur peut survenir occasionnellement en raison de problèmes externes temporaires ou d'erreurs réseau sporadiques. Toutefois, s'il persiste, cela peut indiquer que le proxy de magasin de clés externe ne respecte pas la spécification de l'API du proxy de magasin de clés externe AWS KMS (langue française non garantie). Informez l'administrateur ou le fournisseur de votre magasin de clés externe.

`CustomKeyStoreInvalidStateException` , `KMSInvalidStateException` ou `UnsupportedOperationException`

Le proxy de magasin de clés externe a rejeté la requête, car il ne prend pas en charge l'opération cryptographique demandée.

- Le proxy de stockage de clés externe doit prendre en charge tous les proxys APIs définis dans la spécification de l'API du proxy de stockage de clés AWS KMS externe. Cette erreur indique que le proxy ne prend pas en charge l'opération liée à la requête. Informez l'administrateur ou le fournisseur de votre magasin de clés externe.

Problèmes d'autorisation du proxy

Exceptions : `CustomKeyStoreInvalidStateException`, `KMSInvalidStateException`

Certains proxys de magasin de clés externe mettent en œuvre des exigences d'autorisation pour l'utilisation de leurs clés externes. Un proxy de magasin de clés externe est autorisé, mais pas tenu, de concevoir et d'implémenter un schéma d'autorisation qui permet à des utilisateurs particuliers de demander des opérations particulières sous certaines conditions. Par exemple, un proxy peut autoriser un utilisateur à chiffrer avec une clé externe particulière, mais pas à déchiffrer avec elle.

Pour de plus amples informations, veuillez consulter Autorisation par proxy de magasin de clés externe (facultatif).

L'autorisation du proxy est basée sur les métadonnées AWS KMS incluses dans les demandes adressées au proxy. Les champs `awsSourceVpc` et `awsSourceVpce` ne sont inclus dans les métadonnées que lorsque la requête provient d'un point de terminaison d'un VPC et uniquement lorsque l'appelant possède le même compte que la clé KMS.

```
"requestMetadata": {
    "awsPrincipalArn": string,
    "awsSourceVpc": string, // optional
    "awsSourceVpce": string, // optional
    "kmsKeyArn": string,
    "kmsOperation": string,
    "kmsRequestId": string,
    "kmsViaService": string // optional
}
```

Lorsque le proxy rejette une demande en raison d'un échec d'autorisation, l' AWS KMS opération correspondante échoue. `CreateKey`retours`CustomKeyStoreInvalidStateException`. AWS KMS les opérations cryptographiques retournent`KMSInvalidStateException`. Toutes deux utilisent le message d'erreur suivant :

> Le proxy de magasin de clés externe a refusé l'accès à l'opération. Vérifiez que l'utilisateur et la clé externe sont tous deux autorisés pour cette opération, puis réitérez la requête.

- Pour résoudre l'erreur, utilisez votre gestionnaire de clés externe ou les outils de proxy de votre magasin de clés externe pour déterminer la raison de l'échec de l'autorisation. Ensuite, mettez à jour la procédure à l'origine de la requête non autorisée ou utilisez les outils de proxy de votre magasin de clés externe pour mettre à jour la politique d'autorisation. Vous ne pouvez pas résoudre cette erreur dans AWS KMS.

Sécurité de AWS Key Management Service

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 ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent à AWS Key Management Service (AWS KMS), voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. En plus de votre configuration et de votre utilisation de AWS KMS keys, vous êtes responsable d'autres facteurs, notamment la sensibilité de vos données, les exigences de votre entreprise et les lois et réglementations applicables AWS KMS

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de son utilisation AWS Key Management Service. Il vous explique comment procéder à la configuration AWS KMS pour atteindre vos objectifs de sécurité et de conformité.

Rubriques

Protection des données dans AWS Key Management Service

AWS Key Management Service stocke et protège vos clés de chiffrement afin de les rendre hautement disponibles tout en vous offrant un contrôle d'accès solide et flexible.

Rubriques

Protection des éléments de clé

Par défaut, AWS KMS génère et protège le matériel de clé cryptographique pour les clés KMS. En outre, AWS KMS offre des options pour le matériel clé créé et protégé à l'extérieur de AWS KMS.

Protection du matériel clé généré dans AWS KMS

Lorsque vous créez une clé KMS, par défaut, le matériel cryptographique correspondant à la clé KMS est AWS KMS généré et protégé.

Pour protéger les éléments clés des clés KMS, il AWS KMS s'appuie sur un parc distribué de modules de sécurité matériels validés par la norme FIPS 140-3 de niveau 3 (). HSMs Chaque AWS KMS HSM est une appliance matérielle autonome dédiée conçue pour fournir des fonctions cryptographiques dédiées répondant aux exigences de sécurité et d'évolutivité de. AWS KMS(Les appareils AWS KMS utilisés dans HSMs les régions chinoises sont certifiés par l'OSCCA et sont conformes à toutes les réglementations chinoises pertinentes, mais ne sont pas validés dans le cadre du programme de validation des modules cryptographiques FIPS 140-3.)

L'élément de clé d'une clé KMS est chiffré par défaut lorsqu'il est généré dans le HSM. L'élément de clé est déchiffré uniquement dans la mémoire volatile du HSM et uniquement pendant les quelques millisecondes nécessaires pour l'utiliser dans une opération cryptographique. Chaque fois que le matériel clé n'est pas utilisé activement, il est crypté dans le HSM et transféré vers un stockage persistant hautement durable (99,999999999 %) et à faible latence, où il reste séparé et isolé du. HSMs Les éléments de clé en texte clair ne quittent jamais les limites de sécurité du HSM ; ils ne sont jamais écrits sur disque ou conservés sur un support de stockage. (La seule exception est la clé publique d'une paire de clés asymétriques, qui n'est pas secrète.)

AWS affirme comme principe de sécurité fondamental qu'il n'y a aucune interaction humaine avec les clés cryptographiques en texte clair de quelque type que ce soit. Service AWS Il n'existe aucun mécanisme permettant à quiconque, y compris Service AWS aux opérateurs, de visualiser, d'accéder ou d'exporter du matériel clé en texte brut. Ce principe s'applique même lors de défaillances catastrophiques et d'événements de reprise après sinistre. Le contenu de la clé client en texte brut AWS KMS est utilisé pour les opérations cryptographiques dans le cadre de la AWS KMS norme FIPS 140-3 validée HSMs uniquement en réponse à des demandes autorisées adressées au service par le client ou son délégué.

Pour les clés gérées par le client, la personne Compte AWS qui crée la clé est le propriétaire unique et non transférable de la clé. Le compte propriétaire a un contrôle complet et exclusif sur les politiques d'autorisation qui contrôlent l'accès à la clé. En Clés gérées par AWS effet, ils Compte AWS ont un contrôle total sur les politiques IAM qui autorisent les demandes adressées au Service AWS.

Protection de l’élément de clé généré à l’extérieur de AWS KMS

AWS KMS fournit des alternatives au matériel clé généré dans AWS KMS.

Les magasins de clés personnalisés, une AWS KMS fonctionnalité optionnelle, vous permettent de créer des clés KMS basées sur du matériel clé généré et utilisé en dehors de AWS KMS. Les clés KMS des magasins de AWS CloudHSM clés sont soutenues par des clés des modules de sécurité AWS CloudHSM matériels que vous contrôlez. Ils HSMs sont certifiés au niveau de sécurité 3 de la norme FIPS 140-2 ou au niveau de sécurité 3 de la norme 140-3. Les clés KMS des magasins de clés externes sont soutenues par des clés d'un gestionnaire de clés externe que vous contrôlez et gérez en dehors de AWS celui-ci, tel qu'un HSM physique dans votre centre de données privé.

Une autre fonction facultative vous permet d'importer des éléments de clé pour une clé KMS. Pour protéger le contenu clé importé pendant son transport AWS KMS, vous devez le chiffrer à l'aide d'une clé publique issue d'une paire de clés RSA générée dans un AWS KMS HSM. Le matériel clé importé est déchiffré dans un AWS KMS HSM et rechiffré sous une clé symétrique dans le HSM. Comme tous les documents AWS KMS clés, les éléments clés importés en texte brut ne quittent jamais les éléments HSMs non chiffrés. Cependant, le client qui a fourni les éléments de clé est responsable de l'utilisation en toute sécurité, de la durabilité et de la maintenance des éléments de clé en dehors de AWS KMS.

Chiffrement des données

Les données qu'elles contiennent sont AWS KMS AWS KMS keys constituées du matériel clé de chiffrement qu'elles représentent. Ce matériel clé n'existe en texte clair que dans les modules de sécurité AWS KMS matériels (HSMs) et uniquement lorsqu'il est utilisé. Sinon, les éléments de clé sont chiffrés et stockés dans un stockage permanent durable.

Le matériel clé AWS KMS généré pour les clés KMS ne quitte jamais la limite du AWS KMS HSMs non chiffré. Il n'est ni exporté ni transmis dans le cadre d'aucune opération AWS KMS d'API. L'exception concerne les clés multirégionales, qui AWS KMS utilisent un mécanisme de réplication entre régions pour copier le contenu clé d'une clé multirégionale d'un HSM d'un HSM Région AWS vers un HSM d'un autre. Région AWS Pour plus de détails, voir Processus de réplication pour les clés multirégionales dans Détails AWS Key Management Service cryptographiques.

Rubriques

- Chiffrement au repos
- Chiffrement en transit

Chiffrement au repos

AWS KMS génère des informations clés pour les AWS KMS keys modules de sécurité matériels conformes à la norme FIPS 140-3 Security Level 3 (). HSMs La seule exception concerne les régions chinoises, où les clés KMS sont HSMs AWS KMS utilisées pour générer des clés KMS sont conformes à toutes les réglementations chinoises pertinentes, mais ne sont pas validées dans le cadre du programme de validation des modules cryptographiques FIPS 140-3. Lorsqu'ils ne sont pas utilisés, les éléments de clé sont chiffrés par une clé HSM et écrits dans un stockage permanent et persistant. Le matériel clé pour les clés KMS et les clés de chiffrement qui protègent le contenu clé ne le HSMs quittent jamais sous forme de texte clair.

Le chiffrement et la gestion des éléments de clé pour les clés KMS sont entièrement gérés par AWS KMS.

Pour plus de détails, consultez la section Utilisation AWS KMS keys dans les détails AWS Key Management Service cryptographiques.

Chiffrement en transit

Le matériel clé AWS KMS généré pour les clés KMS n'est jamais exporté ni transmis dans le cadre des opérations AWS KMS d'API. AWS KMS utilise des identificateurs de clé pour représenter les clés

KMS dans les opérations d'API. De même, le contenu clé des clés KMS dans les magasins de clés AWS KMS personnalisés n'est pas exportable et n'est jamais transmis dans le cadre d'opérations AWS KMS d' AWS CloudHSM API.

Cependant, certaines opérations AWS KMS d'API renvoient des clés de données. En outre, les clients peuvent utiliser les opérations d'API pourimporter des éléments de clé pour les clés KMS sélectionnées.

Tous les appels AWS KMS d'API doivent être signés et transmis à l'aide du protocole TLS (Transport Layer Security). AWS KMS nécessite le protocole TLS 1.2 et recommande le protocole TLS 1.3 dans toutes les régions. AWS KMS prend également en charge le protocole TLS post-quantique hybride pour les points de terminaison AWS KMS de service dans toutes les régions, à l'exception des régions de Chine. AWS KMS ne prend pas en charge le protocole TLS post-quantique hybride pour les points de terminaison FIPS dans. AWS GovCloud (US) Les appels vers AWS KMS nécessitent également une suite de chiffrement moderne qui prend en charge une confidentialité persistante parfaite, ce qui signifie que toute violation de secret, telle qu'une clé privée, ne compromet pas également la clé de session.

Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour utiliser des points de AWS KMS terminaison standard ou des points de terminaison AWS KMS FIPS, les clients doivent prendre en charge le protocole TLS 1.2 ou version ultérieure. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3. Pour obtenir la liste des points de terminaison AWS KMS FIPS, consultez la section Points de AWS Key Management Service terminaison et quotas dans le. Références générales AWS

Les communications entre les hôtes du AWS KMS service HSMs sont protégées à l'aide de la cryptographie à courbe elliptique (ECC) et de la norme de chiffrement avancée (AES) dans le cadre d'un schéma de chiffrement authentifié. Pour plus de détails, consultez la section Sécurité des communications internes dans Détails AWS Key Management Service cryptographiques.

Confidentialité du trafic inter-réseaux

AWS KMS prend en charge un AWS Management Console ensemble d'opérations d'API qui vous permettent de les créer, de les gérer AWS KMS keys et de les utiliser dans des opérations cryptographiques.

AWS KMS prend en charge deux options de connectivité réseau allant de votre réseau privé à AWS.

- Une connexion IPSec VPN sur Internet
- AWS Direct Connect, qui relie votre réseau interne à un AWS Direct Connect emplacement via un câble Ethernet à fibre optique standard.

Tous les appels AWS KMS d'API doivent être signés et transmis à l'aide du protocole TLS (Transport Layer Security). Les appels nécessitent également une suite de chiffrement moderne qui prend en charge une confidentialité persistante et parfaite. Le trafic vers les modules de sécurité matériels (HSMs) qui stockent le matériel clé pour les clés KMS est autorisé uniquement à partir d'hôtes d' AWS KMS API connus sur le réseau AWS interne.

Pour vous connecter directement à AWS KMS votre cloud privé virtuel (VPC) sans envoyer de trafic via l'Internet public, utilisez des points de terminaison VPC, alimentés par. AWS PrivateLink Pour de plus amples informations, veuillez consulter Connectez-vous AWS KMS via un point de terminaison VPC.

AWS KMS prend également en charge une option hybride d'échange de clés post-quantique pour le protocole de chiffrement réseau TLS (Transport Layer Security). Vous pouvez utiliser cette option avec le protocole TLS lorsque vous vous connectez aux points de terminaison de AWS KMS l'API.

Gestion des identités et des accès pour AWS Key Management Service

AWS Identity and Access Management (IAM) vous aide à contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les AWS KMS ressources. Pour de plus amples informations, veuillez consulter Utilisation des politiques IAM avec AWS KMS.

Les politiques clés constituent le principal mécanisme de contrôle de l'accès aux clés KMS dans AWS KMS. Chaque clé KMS doit avoir une politique de clé. Vous pouvez également utiliser des stratégies IAM et des octrois, ainsi que des stratégies de clé pour contrôler l'accès à vos clés KMS. Pour de plus amples informations, veuillez consulter Accès aux clés KMS et autorisations.

Si vous utilisez un Amazon Virtual Private Cloud (Amazon VPC), vous pouvez créer un point de terminaison VPC d'interface à utiliser. AWS KMS AWS PrivateLink Vous pouvez également utiliser les politiques de point de terminaison VPC pour déterminer quels principaux peuvent accéder à votre AWS KMS point de terminaison, quels appels d'API ils peuvent effectuer et à quelle clé KMS ils peuvent accéder.

Rubriques

AWS politiques gérées pour AWS Key Management 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 Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

AWS politique gérée : AWSKey ManagementServicePowerUser

Vous pouvez associer la politique `AWSKeyManagementServicePowerUser` à vos identités IAM.

Vous pouvez utiliser une politique gérée par `AWSKeyManagementServicePowerUser` pour accorder aux principaux IAM de votre compte, les autorisations d'un utilisateur avec pouvoir. Les utilisateurs expérimentés peuvent créer des clés KMS, utiliser et gérer les clés KMS qu'ils créent et afficher toutes les clés KMS et les identités IAM. Les principals qui ont la politique gérée `AWSKeyManagementServicePowerUser` peuvent également obtenir des autorisations d'autres sources, notamment des politiques de clé, d'autres politiques IAM et des octrois.

`AWSKeyManagementServicePowerUser`est une politique IAM AWS gérée. Pour plus d'informations sur les politiques AWS gérées, voir les politiques AWS gérées dans le guide de l'utilisateur IAM.

> **Note**
>
> Les autorisations de cette politique qui sont spécifiques à une clé KMS, telles que `kms:TagResource` et `kms:GetKeyRotationStatus`, ne sont effectives que lorsque la politique de clé pour cette clé KMS autorise explicitement le Compte AWS à utiliser des politiques IAM pour contrôler l'accès à la clé. Pour déterminer si une autorisation est spécifique à une clé KMS, consultez AWS KMS autorisations et recherchez une valeur de Clé KMS dans la colonne Ressources.
>
> Cette politique accorde à l'utilisateur expérimenté les autorisations sur n'importe quelle clé KMS avec une politique de clé qui permet l'opération. Pour les autorisations entre comptes, telles que `kms:DescribeKey` et `kms:ListGrants`, cela peut inclure des clés KMS dans des Comptes AWS non approuvés. Pour plus d'informations, consultez Bonnes pratiques pour les politiques IAM et Autoriser des utilisateurs d'autres comptes à utiliser une clé KMS. Pour déterminer si une autorisation est valide sur les clés KMS dans d'autres comptes, consultez AWS KMS autorisations et recherchez la valeur Oui dans la colonne Utilisation inter-comptes.
>
> Pour permettre aux principaux d'accéder à la AWS KMS console sans erreur, ils ont besoin de la balise : GetResources permission, qui n'est pas incluse dans la `AWSKeyManagementServicePowerUser` politique. Vous pouvez accorder cette autorisation dans une politique IAM distincte.

La stratégie IAM gérée par AWSKeyManagementServicePowerUser doit inclure les autorisations suivantes.

- Autorise les principals à créer des clés KMS. Étant donné que ce processus inclut la définition de la politique de clé, les utilisateurs expérimentés peuvent se donner l'autorisation d'utiliser et de gérer les clés KMS qu'ils créent.
- Autorise les principals à créer et supprimer des alias et des balises sur toutes les clés KMS. La modification d'une balise ou d'un alias peut autoriser ou interdire l'utilisation et la gestion de la clé KMS. Pour plus de détails, consultez ABAC pour AWS KMS.
- Permet aux principals d'obtenir des informations détaillées sur toutes les clés KMS, y compris leur ARN de clé, leur configuration de chiffrement, leur politique de clé, leurs alias, leurs balises et leur statut de rotation.
- Permet aux principals de répertorier les utilisateurs, les groupes et les rôles IAM.

- Cette politique n'autorise pas les principals à utiliser ou à gérer des clés KMS qu'ils n'ont pas créées. Cependant, ils peuvent modifier les alias et les balises sur toutes les clés KMS, ce qui peut leur autoriser ou leur refuser l'autorisation d'utiliser ou de gérer une clé KMS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:CreateAlias",
                "kms:CreateKey",
                "kms:DeleteAlias",
                "kms:Describe*",
                "kms:GenerateRandom",
                "kms:Get*",
                "kms:List*",
                "kms:TagResource",
                "kms:UntagResource",
                "iam:ListGroups",
                "iam:ListRoles",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS politique gérée : AWSService RoleForKeyManagementServiceCustomKeyStores

Vous ne pouvez pas joindre de `AWSServiceRoleForKeyManagementServiceCustomKeyStores` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui donne AWS KMS l'autorisation d'afficher les AWS CloudHSM clusters associés à votre magasin de AWS CloudHSM clés et de créer le réseau permettant une connexion entre votre magasin de clés personnalisé et son AWS CloudHSM cluster. Pour de plus amples informations, veuillez consulter Autorisation AWS KMS de gestion AWS CloudHSM et ressources Amazon EC2 .

AWS politique gérée : AWSService RoleForKeyManagementServiceMultiRegionKeys

Vous ne pouvez pas joindre de `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui AWS KMS autorise la synchronisation de toute modification apportée au contenu clé d'une clé primaire multirégionale avec ses clés répliques. Pour de plus amples informations, veuillez consulter Autorisation de synchronisation AWS KMS des clés multirégionales.

AWS KMS mises à jour des politiques AWS gérées

Consultez les détails des mises à jour des politiques AWS gérées AWS KMS depuis que ce service a commencé à suivre ces modifications. Pour obtenir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS de la page AWS KMS Historique du document.

Modification	Description	Date
AWSKeyManagementServiceMultiRegionKeysServiceRolePolicy – Mise à jour de la politique existante	AWS KMS a ajouté un champ Statement ID (`Sid`) à la politique gérée dans la version v2 de la politique.	21 novembre 2024
AWSKeyManagementServiceCustomKeyStoresServiceRolePolicy – Mise à jour de la politique existante	AWS KMS a ajouté les `ec2:DescribeNetworkInterfaces` autorisations `ec2:DescribeVpcs ec2:DescribeNetworkAcls` , et pour surveiller les modifications apportées au VPC qui contient votre AWS CloudHSM cluster afin de AWS KMS pouvoir fournir des messages d'erreur clairs en cas de défaillance.	10 novembre 2023
AWS KMS a commencé à suivre les modifications	AWS KMS a commencé à suivre les modifications	10 novembre 2023

Modification	Description	Date
	apportées AWS à ses politiques gérées.	

Utilisation des rôles liés aux services pour AWS KMS

AWS Key Management Service 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 directement lié à. AWS KMS Les rôles liés au service sont définis par AWS KMS et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service facilite la configuration AWS KMS car il n'est pas nécessaire d'ajouter manuellement les autorisations nécessaires. AWS KMS définit les autorisations associées à ses rôles liés aux services et, sauf indication contraire, seul AWS KMS peut assumer 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 des ressources connexes. Cela protège vos AWS KMS ressources car vous ne pouvez pas supprimer par inadvertance l'autorisation d'accès aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés aux services, consultez Services AWS qui fonctionnent avec IAM et recherchez les services pour lesquels Yes (Oui) est sélectionné dans la colonne Service-Linked Role (Rôle lié aux services). Choisissez un Yes (oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Pour en savoir plus sur les mises à jour des rôles liés aux services abordés dans cette rubrique, consultez. AWS KMS mises à jour des politiques AWS gérées

Rubriques

- Autorisation AWS KMS de gestion AWS CloudHSM et ressources Amazon EC2
- Autorisation de synchronisation AWS KMS des clés multirégionales

Autorisation AWS KMS de gestion AWS CloudHSM et ressources Amazon EC2

Pour prendre en charge vos AWS CloudHSM principaux magasins, vous AWS KMS devez disposer d'une autorisation pour obtenir des informations sur vos AWS CloudHSM clusters. Il a également

besoin d'une autorisation pour créer l'infrastructure réseau qui connecte votre magasin de AWS CloudHSM clés à son AWS CloudHSM cluster. Pour obtenir ces autorisations, AWS KMS crée le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié au service dans votre. Compte AWS Les utilisateurs qui créent des magasins de AWS CloudHSM clés doivent disposer de l'`iam:CreateServiceLinkedRole`autorisation qui leur permet de créer des rôles liés à un service.

Pour obtenir des informations détaillées sur les mises à jour de la politique AWSKeyManagementServiceCustomKeyStoresServiceRolePolicygérée, consultezAWS KMS mises à jour des politiques AWS gérées.

Rubriques

- À propos du rôle lié à AWS KMS un service
- Création du rôle lié à un service
- Modifier la description du rôle lié à un service
- Supprimer le rôle lié à un service

À propos du rôle lié à AWS KMS un service

Un rôle lié à un service est un rôle IAM qui autorise un AWS service à appeler d'autres AWS services en votre nom. Il est conçu pour vous permettre d'utiliser plus facilement les fonctionnalités de plusieurs AWS services intégrés sans avoir à créer et à gérer des politiques IAM complexes. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés aux services pour AWS KMS.

Pour les magasins de AWS CloudHSM clés, AWS KMS crée le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié au service avec la politique AWSKeyManagementServiceCustomKeyStoresServiceRolePolicygérée. Cette politique accorde au rôle les autorisations suivantes :

- cloudHSM:Describe* — détecte les modifications dans le AWS CloudHSM cluster attaché à votre magasin de clés personnalisé.
- ec2 : CreateSecurityGroup — utilisé lorsque vous connectez un magasin de AWS CloudHSM clés pour créer le groupe de sécurité qui permet le flux de trafic réseau entre AWS KMS et votre AWS CloudHSM cluster.
- ec2 : AuthorizeSecurityGroupIngress — utilisé lorsque vous connectez un magasin de AWS CloudHSM clés pour autoriser l'accès au réseau depuis AWS KMS le VPC qui contient AWS CloudHSM votre cluster.

- ec2 : CreateNetworkInterface — utilisé lorsque vous connectez un magasin de AWS CloudHSM clés pour créer l'interface réseau utilisée pour la communication entre AWS KMS et le AWS CloudHSM cluster.
- ec2 : RevokeSecurityGroupEgress — utilisé lorsque vous connectez un magasin de AWS CloudHSM clés pour supprimer toutes les règles sortantes du groupe de sécurité créé. AWS KMS
- ec2 : DeleteSecurityGroup — utilisé lorsque vous déconnectez un magasin de AWS CloudHSM clés pour supprimer les groupes de sécurité créés lors de la connexion du magasin de AWS CloudHSM clés.
- ec2 : DescribeSecurityGroups — utilisé pour surveiller les modifications apportées au groupe de sécurité AWS KMS créé dans le VPC contenant AWS CloudHSM votre cluster afin AWS KMS de fournir des messages d'erreur clairs en cas de défaillance.
- ec2 : DescribeVpcs — utilisé pour surveiller les modifications apportées au VPC qui contient AWS CloudHSM votre cluster afin AWS KMS de pouvoir fournir des messages d'erreur clairs en cas de défaillance.
- ec2 : DescribeNetworkAcls — utilisé pour surveiller les modifications du réseau ACLs pour le VPC qui contient AWS CloudHSM votre cluster afin AWS KMS de pouvoir fournir des messages d'erreur clairs en cas de panne.
- ec2 : DescribeNetworkInterfaces — utilisé pour surveiller les modifications des interfaces réseau AWS KMS créées dans le VPC qui contient AWS CloudHSM votre cluster afin AWS KMS de fournir des messages d'erreur clairs en cas de défaillance.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudhsm:Describe*",
        "ec2:CreateNetworkInterface",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:CreateSecurityGroup",
        "ec2:DescribeSecurityGroups",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:DeleteSecurityGroup",
        "ec2:DescribeVpcs",
        "ec2:DescribeNetworkAcls",
        "ec2:DescribeNetworkInterfaces"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Étant donné que le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié au service n'est fiable que`cks.kms.amazonaws.com`, seul le rôle lié au service AWS KMS peut assumer ce rôle lié au service. Ce rôle est limité aux opérations qui AWS KMS nécessitent de visualiser vos AWS CloudHSM clusters et de connecter un magasin de AWS CloudHSM clés au AWS CloudHSM cluster associé. Il ne donne AWS KMS aucune autorisation supplémentaire. Par exemple, AWS KMS n'est pas autorisé à créer, gérer ou supprimer vos AWS CloudHSM clusters ou vos sauvegardes. HSMs

Régions

À l'instar de la fonctionnalité AWS CloudHSM Key Stores, le AWSServiceRoleForKeyManagementServiceCustomKeyStoresrôle est pris en charge partout Régions AWS où il AWS KMS est disponible. AWS CloudHSM Pour obtenir la liste des points Régions AWS pris en charge par chaque service, voir AWS Key Management Service Points de terminaison et quotas et AWS CloudHSM points de terminaison et quotas dans le. Référence générale d'Amazon Web Services

Pour plus d'informations sur la manière dont les AWS services utilisent les rôles liés aux services, consultez la section Utilisation des rôles liés aux services dans le Guide de l'utilisateur IAM.

Création du rôle lié à un service

AWS KMS crée automatiquement le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié au service dans votre magasin de clés Compte AWS lorsque vous créez un magasin de AWS CloudHSM clés, si le rôle n'existe pas déjà. Vous ne pouvez pas créer ou recréer directement ce rôle lié à un service.

Modifier la description du rôle lié à un service

Vous ne pouvez pas modifier le nom du rôle ou les déclarations de stratégie du rôle lié à un service AWSServiceRoleForKeyManagementServiceCustomKeyStores, mais vous pouvez modifier la description du rôle. Pour obtenir des instructions, veuillez consulter la rubrique Modification d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Supprimer le rôle lié à un service

AWS KMS ne supprime pas le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié au service, Compte AWS même si vous avez supprimé tous vos AWS CloudHSM principaux magasins. Bien qu'il n'existe actuellement aucune procédure permettant de supprimer le rôle AWSServiceRoleForKeyManagementServiceCustomKeyStoreslié à un service, AWS KMS elle n'assume pas ce rôle et n'utilise pas ses autorisations sauf si vous disposez de banques de AWS CloudHSM clés actives.

Autorisation de synchronisation AWS KMS des clés multirégionales

Pour prendre en charge les clés multirégionales, AWS KMS vous devez être autorisé à synchroniser les propriétés partagées d'une clé primaire multirégionale avec ses clés répliquées. Pour obtenir ces autorisations, AWS KMS crée le rôle AWSServiceRoleForKeyManagementServiceMultiRegionKeyslié au service dans votre. Compte AWS Les utilisateurs qui créent des clés multirégionales doivent disposer de l'`iam:CreateServiceLinkedRole`autorisation qui leur permet de créer des rôles liés à un service.

Vous pouvez consulter l'SynchronizeMultiRegionKey CloudTrail événement qui enregistre la AWS KMS synchronisation des propriétés partagées dans vos AWS CloudTrail journaux.

Pour obtenir des informations détaillées sur les mises à jour de la politique AWSKeyManagementServiceMultiRegionKeysServiceRolePolicygérée, consultezAWS KMS mises à jour des politiques AWS gérées.

Rubriques

- À propos du rôle lié à un service pour les clés multi-région
- Création du rôle lié à un service
- Modifier la description du rôle lié à un service
- Supprimer le rôle lié à un service

À propos du rôle lié à un service pour les clés multi-région

Un rôle lié à un service est un rôle IAM qui autorise un AWS service à appeler d'autres AWS services en votre nom. Il est conçu pour vous permettre d'utiliser plus facilement les fonctionnalités de plusieurs AWS services intégrés sans avoir à créer et à gérer des politiques IAM complexes.

Pour les clés multirégionales, AWS KMS crée le rôle AWSServiceRoleForKeyManagementServiceMultiRegionKeyslié au service avec la

AWSKeyManagementServiceMultiRegionKeysServiceRolePolicypolitique gérée. Cette politique donne au rôle l'autorisation `kms:SynchronizeMultiRegionKey`, qui lui permet de synchroniser les propriétés partagées des clés multi-région.

Étant donné que le rôle AWSServiceRoleForKeyManagementServiceMultiRegionKeyslié au service n'est fiable que`mrk.kms.amazonaws.com`, seul le rôle lié au service AWS KMS peut assumer ce rôle lié au service. Ce rôle est limité aux opérations qui AWS KMS doivent synchroniser les propriétés partagées multirégionales. Il ne donne AWS KMS aucune autorisation supplémentaire. Par exemple, AWS KMS n'est pas autorisé à créer, répliquer ou supprimer des clés KMS.

Pour plus d'informations sur la manière dont les AWS services utilisent les rôles liés aux services, consultez la section Utilisation des rôles liés aux services dans le guide de l'utilisateur IAM.

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Sid" : "KMSSynchronizeMultiRegionKey",
            "Effect" : "Allow",
            "Action" : [
                "kms:SynchronizeMultiRegionKey"
            ],
            "Resource" : "*"
        }
    ]
}
```

Création du rôle lié à un service

AWS KMS crée automatiquement le rôle AWSServiceRoleForKeyManagementServiceMultiRegionKeyslié au service dans votre Compte AWS lorsque vous créez une clé multirégionale, si le rôle n'existe pas déjà. Vous ne pouvez pas créer ou recréer directement ce rôle lié à un service.

Modifier la description du rôle lié à un service

Vous ne pouvez pas modifier le nom du rôle ni les déclarations de politique du rôle AWSServiceRoleForKeyManagementServiceMultiRegionKeyslié à un service, mais vous pouvez modifier la description du rôle. Pour de plus amples informations, veuillez consulter Modification d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Supprimer le rôle lié à un service

AWS KMS ne supprime pas le rôle AWSServiceRoleForKeyManagementServiceMultiRegionKeyslié au service de votre compte Compte AWS et vous ne pouvez pas le supprimer. Cependant, AWS KMS n'assume pas le AWSServiceRoleForKeyManagementServiceMultiRegionKeysrôle et n'utilise aucune de ses autorisations, sauf si vous avez des clés multirégionales dans votre région Compte AWS et.

Connexion et surveillance AWS Key Management Service

La surveillance est un élément important permettant de comprendre la disponibilité, l'état et l'utilisation de vos AWS KMS keys dans AWS KMS. La surveillance permet de maintenir la sécurité, la fiabilité, la disponibilité et les performances de vos AWS solutions. AWS fournit plusieurs outils pour surveiller vos clés KMS.

AWS CloudTrail Journaux

Chaque appel à une opération d' AWS KMS API est enregistré sous forme d'événement dans un AWS CloudTrail journal. Ces journaux enregistrent tous les appels d'API depuis la AWS KMS console, ainsi que les appels effectués par AWS KMS d'autres AWS services. Les appels d'API entre comptes, tels qu'un appel à utiliser une clé KMS dans un autre compte Compte AWS, sont enregistrés dans les CloudTrail journaux des deux comptes.

Lors du dépannage ou de l'audit, vous pouvez utiliser le journal pour reconstruire le cycle de vie d'une clé KMS. Vous pouvez également afficher sa gestion et son utilisation de la clé KMS dans les opérations de chiffrement. Pour de plus amples informations, veuillez consulter the section called “Se connecter avec AWS CloudTrail”.

Amazon CloudWatch Logs

Surveillez, stockez et accédez à vos fichiers journaux depuis AWS CloudTrail d'autres sources. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.

Car AWS KMS, CloudWatch stocke des informations utiles qui vous aident à éviter les problèmes liés à vos clés KMS et aux ressources qu'elles protègent. Pour de plus amples informations, veuillez consulter the section called “Touches du moniteur avec CloudWatch”.

Amazon EventBridge

AWS KMS génère EventBridge des événements lorsque votre clé KMS est pivotée ou supprimée ou lorsque le contenu clé importé de votre clé KMS expire. Recherchez AWS KMS des

événements (opérations d'API) et acheminez-les vers une ou plusieurs fonctions ou flux cibles pour capturer des informations d'état. Pour plus d'informations, consultez the section called “Surveillez les touches avec Amazon EventBridge” le guide de EventBridge l'utilisateur Amazon.

CloudWatch Métriques Amazon

Vous pouvez surveiller vos clés KMS à l'aide de CloudWatch métriques, qui collectent et traitent les données brutes pour AWS KMS en faire des indicateurs de performance. Les données sont enregistrées à intervalles de deux semaines afin que vous puissiez visualiser les tendances des informations actuelles et historiques. Cela vous aide à comprendre comment vos clés KMS sont utilisées et comment leur utilisation évolue au fil du temps. Pour plus d'informations sur l'utilisation de CloudWatch métriques pour surveiller les clés KMS, consultezAWS KMS métriques et dimensions.

CloudWatch Alarmes Amazon

Surveillez les évolutions d'une seule métrique pendant la période que vous spécifiez. Ensuite, prenez des mesures en fonction de la valeur de la métrique par rapport à un seuil sur un certain nombre de périodes. Par exemple, vous pouvez créer une CloudWatch alarme qui se déclenche lorsque quelqu'un essaie d'utiliser une clé KMS dont la suppression est programmée dans le cadre d'une opération cryptographique. Cela indique que la clé KMS est toujours utilisée et ne devrait probablement pas être supprimée. Pour de plus amples informations, veuillez consulter the section called “Créer une alarme”.

AWS Security Hub

Vous pouvez surveiller votre AWS KMS utilisation conformément aux normes du secteur de la sécurité et aux meilleures pratiques en utilisant AWS Security Hub. Security Hub utilise des contrôles de sécurité pour évaluer les configurations des ressources et les normes de sécurité afin de vous aider à respecter divers cadres de conformité. Pour plus d'informations, consultez Concepts AWS Key Management Service dans le Guide de l'utilisateur AWS Security Hub .

Validation de conformité pour AWS Key Management Service

Des auditeurs tiers évaluent la sécurité et AWS Key Management Service la conformité de plusieurs programmes de AWS conformité. Il s’agit notamment des certifications SOC, PCI, FedRAMP, HIPAA et d’autres.

Rubriques

- Documents de conformité et de sécurité

- En savoir plus

Documents de conformité et de sécurité

Les documents de conformité et de sécurité suivants couvrent AWS KMS. Pour ce faire, utilisez AWS Artifact.

- Cloud Computing Compliance Controls Catalogue (C5)
- ISO 27001 : Déclaration d'applicabilité 2013 (DdA)
- ISO 27001 : Certification 2013
- ISO 27017 : Déclaration d'applicabilité 2015 (DdA)
- ISO 27017 : Certification 2015
- ISO 27018 : Déclaration d'applicabilité 2015 (DdA)
- ISO 27018 : Certification 2014
- ISO 9001: Certification 2015
- Attestation de conformité (AOC) et récapitulatif des responsabilités PCI DSS
- Rapport SOC 1 (Service Organization Controls)
- Rapport SOC 2 (Service Organization Controls)
- Rapport SOC 2 (Service Organization Controls) relatif à la confidentialité
- FedRAMP-High

Pour obtenir de l'aide sur l'utilisation AWS Artifact, consultez la section Téléchargement de rapports dans AWS Artifact.

En savoir plus

Votre responsabilité en matière de conformité lors de l'utilisation AWS KMS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. Si votre utilisation de AWS KMS est soumise à la conformité à une norme publiée, AWS fournit des ressources pour vous aider à :

- AWS Services concernés par programme de conformité — Cette page répertorie les AWS services concernés par des programmes de conformité spécifiques. Pour obtenir des informations générales, consultez Programmes de conformitéAWS.

- 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 la sécurité et la conformité sur. AWS
- AWS Ressources relatives à la conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- AWS Config— Ce AWS 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— Ce AWS service fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos AWS ressources et vérifier votre conformité aux normes et aux meilleures pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.

Résilience dans AWS Key Management Service

L'infrastructure AWS mondiale est construite autour Régions AWS de zones de disponibilité. Régions AWS fournissent plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone 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.

Outre l'infrastructure AWS mondiale, AWS KMS propose plusieurs fonctionnalités pour répondre à vos besoins en matière de résilience et de sauvegarde des données. Pour plus d'informations sur les zones de disponibilité Régions AWS et les zones de disponibilité, consultez la section Infrastructure AWS globale.

Isolement régional

AWS Key Management Service (AWS KMS) est un service régional autonome qui est disponible partout Régions AWS. La conception isolée au niveau régional de AWS KMS garantit qu'un problème de disponibilité dans une région Région AWS ne peut affecter le AWS KMS fonctionnement dans aucune autre région. AWS KMS est conçu pour garantir l'absence d'interruption planifiée, toutes les mises à jour logicielles et les opérations de dimensionnement étant effectuées de manière fluide et imperceptible.

Le contrat AWS KMS de niveau de service (SLA) inclut un engagement de service de 99,999 % pour tous les KMS. APIs Pour remplir cet engagement, AWS KMS garantit que toutes les données et informations d'autorisation nécessaires à l'exécution d'une requête d'API sont disponibles sur tous les hôtes régionaux qui reçoivent la requête.

L' AWS KMS infrastructure est répliquée dans au moins trois zones de disponibilité (AZs) dans chaque région. Afin de garantir que les défaillances de plusieurs hôtes n'affectent pas les AWS KMS performances, AWS KMS il est conçu pour gérer le trafic client AZs en provenance de n'importe quelle région.

Les modifications que vous apportez aux propriétés ou aux autorisations d'une clé KMS sont répliquées pour tous les hôtes de la région afin de garantir que la requête ultérieure peut être traitée correctement par n'importe quel hôte de la région. Les demandes d'opérations cryptographiques utilisant votre clé KMS sont transmises à un parc de modules de sécurité AWS KMS matériels (HSMs), chacun d'entre eux pouvant effectuer l'opération avec la clé KMS.

Conception à locataires multiples

La conception multi-locataires de AWS KMS lui permet de respecter le SLA de disponibilité de 99,999 % et de maintenir des taux de demandes élevés, tout en protégeant la confidentialité de vos clés et de vos données.

Plusieurs mécanismes renforçant l'intégrité sont déployés pour garantir que la clé KMS que vous avez spécifiée pour l'opération cryptographique est toujours celle utilisée.

Les éléments de clé en texte clair de vos clés KMS sont protégés de manière stricte. Les éléments de clé sont chiffrés dans le HSM dès sa création, et les éléments de clé chiffrés sont immédiatement déplacés vers un stockage sécurisé à faible latence. La clé chiffrée est récupérée et déchiffrée dans le HSM juste à temps pour être utilisée. La clé en texte clair reste dans la mémoire HSM uniquement pendant le temps nécessaire à la réalisation de l'opération cryptographique. Il est ensuite rechiffré dans le HSM et la clé chiffrée est renvoyée au stockage. Le contenu clé en texte brut ne quitte jamais le HSMs ; il n'est jamais écrit dans un stockage permanent.

Meilleures pratiques en matière de résilience dans AWS KMS

Pour optimiser la résilience de vos AWS KMS ressources, envisagez les stratégies suivantes.

- Pour prendre en charge votre stratégie de sauvegarde et de reprise après sinistre, prenez en compte les Clés multi-régions, qui sont des clés KMS créées dans une Région AWS et répliquées uniquement pour les régions que vous spécifiez. Avec les clés multirégionales, vous pouvez

déplacer des ressources chiffrées entre Régions AWS (au sein d'une même partition) sans jamais exposer le texte en clair, et déchiffrer la ressource, si nécessaire, dans l'une de ses régions de destination. Les clés multi-régions associées sont interopérables car elles partagent les mêmes éléments de clé et le même ID de clé, mais elles disposent de stratégies clé indépendantes pour le contrôle d'accès haute résolution. Pour plus de détails, veuillez consulter Clés multi-régions dans AWS KMS.

- Pour protéger vos clés dans un service mutualisé tel que AWS KMS, veillez à utiliser des contrôles d'accès, notamment des politiques clés et des politiques IAM. En outre, vous pouvez envoyer vos demandes à AWS KMS l'aide d'un point de terminaison d'interface VPC alimenté par. AWS PrivateLink Lorsque vous le faites, toutes les communications entre votre Amazon VPC et Amazon AWS KMS sont entièrement effectuées au sein du AWS réseau à l'aide d'un point de AWS KMS terminaison dédié limité à votre VPC. Vous pouvez sécuriser davantage ces requêtes en créant une couche d'autorisation supplémentaire à l'aide des Stratégies de point de terminaison d'un VPC. Pour en savoir plus, consultez la section Connexion à AWS KMS via un point de terminaison d'un VPC.

Sécurité de l'infrastructure dans AWS Key Management Service

En tant que service géré, AWS Key Management Service (AWS KMS) est protégé par les procédures de sécurité du réseau AWS mondial décrites dans Amazon Web Services : présentation des processus de sécurité.

Pour accéder AWS KMS via le réseau, vous pouvez appeler les opérations d' AWS KMS API décrites dans la référence AWS Key Management Service d'API. AWS KMS nécessite le protocole TLS 1.2 et recommande le protocole TLS 1.3 dans toutes les régions. AWS KMS prend également en charge le protocole TLS post-quantique hybride pour les points de terminaison AWS KMS de service dans toutes les régions, à l'exception des régions de Chine. AWS KMS ne prend pas en charge le protocole TLS post-quantique hybride pour les points de terminaison FIPS dans. AWS GovCloud (US) Pour utiliser les points de terminaison AWS KMS standard ou les points de terminaison FIPS AWS KMS, les clients doivent prendre en charge le protocole TLS 1.2 ou une version ultérieure. Les clients doivent aussi prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La plupart des systèmes modernes telles que Java 7 et 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.

Vous pouvez appeler ces opérations d'API depuis n'importe quel emplacement réseau, mais AWS KMS les conditions de politique globale vous permettent de contrôler l'accès à une clé KMS en fonction de l'adresse IP source, du VPC et du point de terminaison du VPC. Vous pouvez utiliser ces clés de condition dans les stratégies de clé et les stratégies IAM. Toutefois, ces conditions peuvent AWS empêcher l'utilisation de la clé KMS en votre nom. Pour plus de détails, consultez AWS clés de condition globales.

Par exemple, la déclaration de politique clé suivante permet aux utilisateurs qui peuvent assumer le `KMSTestRole` rôle de l'utiliser AWS KMS key pour les opérations cryptographiques spécifiées, sauf si l'adresse IP source est l'une des adresses IP spécifiées dans la stratégie.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS":
    "arn:aws:iam::111122223333:role/KMSTestRole"},
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt",
      "kms:ReEncrypt*",
      "kms:GenerateDataKey*",
      "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
      "NotIpAddress": {
        "aws:SourceIp": [
          "192.0.2.0/24",
          "203.0.113.0/24"
        ]
      }
    }
  }
}
```

Isolation des hôtes physiques

La sécurité de l'infrastructure physique AWS KMS utilisée est soumise aux contrôles décrits dans la section Sécurité physique et environnementale d'Amazon Web Services : présentation des processus de sécurité. Vous trouverez plus de détails dans les rapports de conformité et les résultats d'audit tiers répertoriés dans la section précédente.

AWS KMS est pris en charge par des modules de sécurité matériels renforcés dédiés (HSMs) conçus avec des contrôles spécifiques pour résister aux attaques physiques. HSMs Il s'agit de périphériques physiques dépourvus de couche de virtualisation, telle qu'un hyperviseur, qui partage le périphérique physique entre plusieurs locataires logiques. Le matériel clé pour AWS KMS keys est stocké uniquement dans la mémoire volatile du HSMs, et uniquement pendant l'utilisation de la clé KMS. Cette mémoire est effacée lorsque le HSM sort de l'état opérationnel, y compris lors des arrêts et des réinitialisations prévus et involontaires. Pour des informations détaillées sur le fonctionnement de AWS KMS HSMs, voir Détails AWS Key Management Service cryptographiques.

Quotas

Pour garantir la AWS KMS réactivité et les performances de tous les utilisateurs, AWS KMS appliquez deux types de quotas : les quotas de ressources et les quotas de demande. Chaque quota est calculé indépendamment pour chaque région de chaque Compte AWS.

Tous les AWS KMS quotas sont ajustables, à l'exception du quota de ressources de rotation à la demande et du quota de demandes de stockage de AWS CloudHSM clés. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas. Pour demander une réduction de quota, pour modifier un quota qui ne figure pas dans la liste des Quotas de Service, ou pour modifier un quota dans un pays Région AWS où les Quotas de Service pour AWS KMS ne sont pas disponibles, rendez-vous AWS Support au Centre et créez un dossier.

Rubriques

Quotas de ressources

AWS KMS établit des quotas de ressources pour garantir qu'il peut fournir un service rapide et résilient à tous nos clients. Certains quotas de ressources s'appliquent uniquement aux ressources que vous créez, mais pas aux ressources que les AWS services créent pour vous. Les ressources que vous utilisez, mais qui ne sont pas dans votre Compte AWS, telles que les Clés détenues par AWS, ne sont pas prises en compte dans le calcul de ces quotas.

Si vous avez atteint une limite de ressources, les demandes de création d'une ressource supplémentaire de ce type génèrent un message d'erreur `LimitExceededException`.

Tous les quotas de AWS KMS ressources sont ajustables, à l'exception du quota de ressources de rotation à la demande. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas. Pour demander une réduction de quota, pour modifier un quota qui ne figure pas dans la liste des Quotas de Service, ou pour modifier un quota dans un pays Région AWS où les Quotas de Service pour AWS KMS ne sont pas disponibles, rendez-vous AWS Support au Centre et créez un dossier.

Le tableau suivant répertorie et décrit les quotas de AWS KMS ressources dans chaque Compte AWS région.

Nom du quota	Valeur par défaut	S'applique à	Ajustable
AWS KMS keys	100 000	Clés gérées par le client	Oui
Alias par clé KMS	50	Alias créés par le client	Oui
Octrois par clé KMS	50 000	Clés gérées par le client	Oui
Quota de ressources du magasin de clé personnalisé	10	Compte AWS et région	Oui
Rotation à la demande	10	Clés gérées par le client	Non

Outre les quotas de ressources, AWS KMS utilise des quotas de demande pour garantir la réactivité du service. Pour plus de détails, consultez the section called "Quotas de demande".

AWS KMS keys :100 000

Vous pouvez avoir jusqu'à 100 000 clés gérées par le client dans chaque région de votre Compte AWS. Ce quota s'applique à toutes les clés gérées par le client dans toutes les Régions AWS indépendamment de leur Spécification de clé ou état de clé. Chaque clé KMS est considérée comme une ressource unique. Les Clés gérées par AWS et Clés détenues par AWS ne sont pas prises en compte dans ce quota.

Alias par clé KMS : 50

Vous pouvez associer jusqu'à 50 alias avec chaque clé gérée par le client. Les alias AWS associés à ne sont Clés gérées par AWSpas pris en compte dans ce quota. Vous pouvez rencontrer ce quota lorsque vous créez ou mettez à jour un alias.

> **Note**
>
> La ResourceAliases condition kms : n'est effective que lorsque la clé KMS est conforme à ce quota. Si une clé KMS dépasse ce quota, les mandataires autorisés à utiliser la clé KMS par la condition `kms:ResourceAliases` se voient refuser l'accès à la clé KMS. Pour plus de détails, consultez Accès refusé en raison d'un quota d'alias.

Le quota d'alias par clé KMS remplace le quota d'alias par région qui limitait le nombre total d'alias dans chaque région d'un. Compte AWS AWS KMS a éliminé le quota d'alias par région.

Octrois par clé KMS : 50 000

Chaque clé gérée par le client peut avoir jusqu'à 50 000 octrois, y compris les octrois créés par les services AWS qui sont intégrés à AWS KMS. Ce quota ne s'applique pas aux Clés gérées par AWS ou Clés détenues par AWS.

L'un des effets de ce quota est que vous ne pouvez pas exécuter plus de 50 000 opérations autorisées par octroi qui utilisent simultanément la même clé KMS. Une fois que vous avez atteint le quota, vous pouvez créer de nouveaux octrois sur la clé KMS uniquement lorsqu'un octroi actif est retiré ou révoqué.

Par exemple, lorsque vous associez un volume Amazon Elastic Block Store (Amazon EBS) à une instance Amazon Elastic Compute Cloud (EC2Amazon), le volume est déchiffré afin que vous puissiez le lire. Pour obtenir l'autorisation de déchiffrer les données, Amazon EBS crée un octroi pour chaque volume. Par conséquent, si tous vos volumes Amazon EBS utilisent la même clé KMS, vous ne pouvez pas attacher plus de 50 000 volumes en même temps.

Quota de ressources des magasins de clés personnalisés : 10

Vous pouvez créer jusqu'à 10 magasins de clés personnalisés dans chaque Compte AWS région. Si vous essayez d'en créer d'autres, l'CreateCustomKeyStoreopération échoue.

Ce quota s'applique au nombre total de magasins de clés personnalisés dans chaque compte et région, y compris tous les magasins de clés AWS CloudHSM et les magasins de clés externes, quel que soit leur état de connexion.

Rotation à la demande : 10

Vous pouvez effectuer une rotation de clé à la demande au maximum 10 fois par clé KMS. Si vous essayez d'effectuer davantage de rotations à la demande, l'RotateKeyOnDemandopération échoue.

Il ne s'agit pas d'un quota ajustable. Vous ne pouvez pas l'augmenter en utilisant des Quotas de Service ou en créant un dossier AWS Support. Pour éviter d'atteindre le quota de rotation à la demande, nous vous recommandons d'utiliser la rotation automatique des touches dans la mesure du possible.

Quotas de demande

AWS KMS établit des quotas pour le nombre d'opérations d'API demandées par seconde. Les quotas de demandes varient en fonction du fonctionnement de l'API Région AWS, du et d'autres facteurs, tels que le type de clé KMS. Lorsque vous dépassez un quota de demandes d'API, AWS KMS la demande est limitée.

Tous les quotas de AWS KMS demandes sont ajustables, à l'exception du quota de demandes du magasin de AWS CloudHSM clés. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas. Pour demander une réduction de quota, pour modifier un quota qui ne figure pas dans la liste des Quotas de Service, ou pour modifier un quota dans un pays Région AWS où les Quotas de Service pour AWS KMS ne sont pas disponibles, rendez-vous AWS Support au Centre et créez un dossier.

Si vous dépassez le quota de demandes pour l'GenerateDataKeyopération, pensez à utiliser la fonctionnalité de mise en cache des clés de données du AWS Encryption SDK. La réutilisation des clés de données peut réduire la fréquence de vos demandes de AWS KMS.

Outre les quotas de demande, AWS KMS utilise des quotas de ressources pour garantir la capacité de tous les utilisateurs. Pour en savoir plus, consultez Quotas de ressources.

Pour afficher les tendances de vos taux de demande, utilisez la console Service Quotas. Vous pouvez également créer une CloudWatch alarme Amazon qui vous avertit lorsque le taux de demandes atteint un certain pourcentage de la valeur du quota. Pour plus de détails, consultez Gérer vos taux de demandes AWS KMS d'API à l'aide de Service Quotas et d'Amazon CloudWatch dans le blog sur la AWS sécurité.

Rubriques

- Quotas de demande pour chaque opération AWS KMS d'API

- Application des quotas de demande
- Quotas partagés pour les opérations de chiffrement
- Demandes d'API effectuées en votre nom
- Demandes entre comptes
- Quotas de demandes de magasin de clés personnalisé

Quotas de demande pour chaque opération AWS KMS d'API

Ce tableau répertorie le code de quota de Service Quotas et la valeur par défaut pour chaque quota de AWS KMS demande. Tous les quotas de AWS KMS demandes sont ajustables, à l'exception du quota de demandes du magasin de AWS CloudHSM clés.

> **Note**
>
> Il peut être nécessaire de faire défiler horizontalement ou verticalement pour afficher toutes les données de ce tableau.

Nom du quota	Valeur par défaut (requêtes par seconde)

`Cryptographic operations (symmetric) request rate`
 	
S'applique à :
 	
• `Decrypt`
• `Encrypt`
• `GenerateDataKey`
• `GenerateDataKeyWithoutPlaintext`
• `GenerateMac`
• `GenerateRandom`
• `ReEncrypt`
• `VerifyMac`
	Ces quotas partagés varient en fonction de la clé KMS utilisée dans la demande Région AWS et du type de clé KMS. Chaque quota est calculé séparément.
	
	• 10 000 (partagés)
	• 20 000 (partagés) dans les régions suivantes :
	  • USA Est (Ohio), us-east-2
	  • Asie-Pacifique (Singapour), ap-southeast-1
	  • Asie-Pacifique (Sydney), ap-southeast-2
	  • Asie-Pacifique (Tokyo), ap-northeast-1
	  • Europe (Francfort), eu-central-1
	  • Europe (Londres), eu-west-2

Nom du quota	Valeur par défaut (requêtes par seconde)

	• 100 000 (partagés) dans les régions suivantes :
	  • USA Est (Virginie du Nord), us-east-1
	  • USA Ouest (Oregon), us-west-2
	  • Europe (Irlande), eu-west-1

`Cryptographic operations (RSA) request rate` S'applique à : • `Decrypt` • `Encrypt` • `ReEncrypt` • `Sign` • `Verify`	1 000 (partagés) pour les clés RSA KMS
`Cryptographic operations (ML-DSA) request rate` S'applique à : • `Sign` • `Verify`	1 000 (partagés) pour les clés KMS ML-DSA

Nom du quota	Valeur par défaut (requêtes par seconde)
`Cryptographic operations (ECC and SM2) request rate` S'applique à : • `Decrypt`—uniquement pris en charge pour les SM2 clés KMS (régions de Chine uniquement) • `DeriveSharedSecret` • `Encrypt`—uniquement pris en charge pour les SM2 clés KMS (régions de Chine uniquement) • `ReEncrypt` —uniquement pris en charge pour les SM2 clés KMS (régions de Chine uniquement) • `Sign` • `Verify`	1 000 (partagées) pour les clés KMS à courbe elliptique (ECC) et (régions de SM2 Chine uniquement)
`Custom key store request quotas` S'applique à : • `Decrypt` • `DeriveSharedSecret` • `Encrypt` • `GenerateDataKey` • `GenerateDataKeyWithoutPlaintext` • `GenerateRandom` • `ReEncrypt`	Les quotas de demandes de magasin de clés personnalisé sont calculés séparément pour chaque magasin de clés personnalisé • 1 800 (partagés) pour chaque magasin de AWS CloudHSM clés • 1 800 (partagées) pour chaque magasin de clés externes
`CancelKeyDeletion request rate`	5

Nom du quota	Valeur par défaut (requêtes par seconde)
ConnectCustomKeyStore request rate	5
CreateAlias request rate	5
CreateCustomKeyStore request rate	5
CreateGrant request rate	50
CreateKey request rate	5
DeleteAlias request rate	15
DeleteCustomKeyStore request rate	5
DeleteImportedKeyMaterial request rate	15
DescribeCustomKeyStores request rate	5
DescribeKey request rate	2000
DisableKey request rate	5
DisableKeyRotation request rate	5
DisconnectCustomKeyStore request rate	5
EnableKey request rate	5
EnableKeyRotation request rate	15

Nom du quota	Valeur par défaut (requêtes par seconde)
`GenerateDataKeyPair (ECC_NIST_P256) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	100
`GenerateDataKeyPair (ECC_NIST_P384) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	100
`GenerateDataKeyPair (ECC_NIST_P521) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	100
`GenerateDataKeyPair (ECC_SECG_P256K1) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	100

Nom du quota	Valeur par défaut (requêtes par seconde)
`GenerateDataKeyPair (RSA_2048) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	1
`GenerateDataKeyPair (RSA_3072) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	0,5 (1 dans chaque intervalle de 2 secondes)
`GenerateDataKeyPair (RSA_4096) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	0,1 (1 dans chaque intervalle de 10 secondes)
`GenerateDataKeyPair (SM2 – China Regions only) request rate` S'applique à : • `GenerateDataKeyPair` • `GenerateDataKeyPairWithoutPlaintext`	25
`GetKeyPolicy request rate`	1 000

Nom du quota	Valeur par défaut (requêtes par seconde)
`GetKeyRotationStatus request rate`	1 000
`GetParametersForImport request rate`	0,25 (1 dans chaque intervalle de 4 secondes)
`GetPublicKey request rate`	2000
`ImportKeyMaterial request rate`	15
`ListAliases request rate`	500
`ListGrants request rate`	100
`ListKeyPolicies request rate`	100
`ListKeys request rate`	500
`ListKeyRotations request rate`	100
`ListResourceTags request rate`	2000
`ListRetirableGrants request rate`	100
`PutKeyPolicy request rate`	15
`ReplicateKey request rate` Une opération `ReplicateKey` compte comme une demande `ReplicateKey` dans la région de la clé principale et deux demandes `CreateKey` dans la région du réplica. L'une des demandes `CreateKey` est un essai pour détecter les problèmes potentiels avant de créer la clé.	5
`RetireGrant request rate`	50
`RevokeGrant request rate`	50

Nom du quota	Valeur par défaut (requêtes par seconde)
`RotateKeyOnDemand request rate`	5
`ScheduleKeyDeletion request rate`	15
`TagResource request rate`	10
`UntagResource request rate`	5
`UpdateAlias request rate`	5
`UpdateCustomKeyStore request rate`	5
`UpdateKeyDescription request rate`	5
`UpdatePrimaryRegion request rate` Une opération `UpdatePrimaryRegion` compte comme deux demandes `UpdatePrimaryRegion` ; une demande dans chacune des deux régions touchées.	5

Application des quotas de demande

Lors de la révision des quotas de demande, gardez à l'esprit les informations suivantes.

- Les quotas de demande s'appliquent aux clés gérées par le client et aux Clés gérées par AWS. L'utilisation de Clés détenues par AWSn'est pas prise en compte dans les quotas de demandes pour vous Compte AWS, même s'ils sont utilisés pour protéger les ressources de votre compte.
- Les quotas de demande s'appliquent aux demandes envoyées aux points de terminaison FIPS et aux points de terminaison non FIPS. Pour obtenir la liste des points de terminaison de AWS KMS service, consultez la section AWS Key Management Service Points de terminaison et quotas dans le. Références générales AWS
- La limitation est basée sur toutes les demandes concernant les clés KMS de tous types dans la région. Ce total inclut les demandes émanant de tous les principaux acteurs du Compte AWS, y compris les demandes émanant de AWS services en votre nom.

- Chaque quota de demande est calculé indépendamment. Par exemple, les demandes relatives à l'CreateKeyopération n'ont aucun effet sur le quota de demandes pour l'CreateAliasopération. Si vos demandes `CreateAlias` sont soumises à une limitation, vos demandes `CreateKey` peuvent tout de même s'exécuter avec succès.
- Bien que les opérations de chiffrement partagent un quota, le quota partagé est calculé indépendamment des quotas appliqués aux autres opérations. Par exemple, les appels aux opérations de chiffrement et de déchiffrement partagent un quota de demandes, mais ce quota est indépendant du quota pour les opérations de gestion, telles que. EnableKey Par exemple, dans la région Europe (Londres), vous pouvez effectuer 10 000 opérations de chiffrement sur des clés KMS symétriques plus 5 opérations `EnableKey` par seconde sans être limité.

Quotas partagés pour les opérations de chiffrement

AWS KMS les opérations cryptographiques partagent les quotas de demandes. Vous pouvez demander n'importe quelle combinaison d'opérations de chiffrement prises en charge par la clé KMS, à condition que le nombre total d'opérations de chiffrement ne dépasse pas le quota de demande pour ce type de clé KMS. Les exceptions sont GenerateDataKeyPairet GenerateDataKeyPairWithoutPlaintext, qui partagent un quota distinct.

Les quotas des différents types de clés KMS sont calculés indépendamment. Chaque quota s'applique à toutes les demandes relatives à ces opérations dans la région Compte AWS et avec le type de clé donné à chaque intervalle d'une seconde.

- Le taux de demande des opérations de chiffrement (symétrique) est le quota de demande partagé pour les opérations de chiffrement utilisant des clés KMS symétriques dans un compte et une région. Ce quota s'applique aux opérations cryptographiques avec des clés de chiffrement symétriques et des clés HMAC, qui sont également symétriques.

  Par exemple, vous pouvez utiliser des clés KMS symétriques Région AWS avec un quota partagé de 10 000 demandes par seconde. Lorsque vous faites 7 000 GenerateDataKeydemandes par seconde et 2 000 demandes de déchiffrement par seconde, AWS KMS cela ne limite pas vos demandes. Par contre, si vous effectuez 9 500 demandes `GenerateDataKey` et 1 000 demandes Encrypt par seconde, AWS KMS limite vos demandes, car elles dépassent le quota partagé.

  Les opérations de chiffrement sur les clés KMS de chiffrement symétrique d'un magasin de clés personnalisé sont à la fois comptabilisées dans le taux de demandes d'opérations de chiffrement (symétrique) du compte et dans le quota de demandes du magasin de clés personnalisé.

- Le taux de demande RSA (opérations de chiffrement) est le quota de demande partagé pour les opérations de chiffrement utilisant des clés KMS asymétriques RSA.

  Par exemple, avec un quota de 1 000 opérations par seconde, vous pouvez effectuer 400 demandes de chiffrement et 200 demandes de déchiffrement avec des clés RSA KMS capables de chiffrer et de déchiffrer, ainsi que 250 demandes de signature et 150 demandes de vérification avec des clés RSA KMS capables de signer et de vérifier.
- Le taux de demandes d'opérations cryptographiques (ECC) est le quota de demandes partagé pour les opérations cryptographiques utilisant des clés KMS asymétriques à courbe elliptique (ECC) et des clés KMS asymétriques SM.

  Par exemple, avec un quota de 1 000 opérations par seconde, vous pouvez effectuer 400 demandes de signature et 200 demandes de vérification avec des clés KMS ECC capables de signer et de vérifier, plus 250 demandes de signature et 150 demandes de vérification avec des clés SM2 KMS capables de signer et de vérifier.
- Le quota de demandes du magasin de clés personnalisé désigne le quota de demandes partagées pour les opérations de chiffrement sur les clés KMS d'un magasin de clés personnalisé. Cette limite est calculée séparément pour chaque magasin de clés personnalisé.

  Les opérations de chiffrement sur les clés KMS de chiffrement symétrique d'un magasin de clés personnalisé sont à la fois comptabilisées dans le taux de demandes d'opérations de chiffrement (symétrique) du compte et dans le quota de demandes du magasin de clés personnalisé.

Les quotas des différents types de clés sont également calculées indépendamment. Par exemple, dans la région Asie-Pacifique (Singapour), si vous utilisez à la fois des clés KMS symétriques et asymétriques, vous pouvez effectuer jusqu'à 10 000 appels par seconde avec des clés KMS symétriques (y compris des clés HMAC) plus 500 appels supplémentaires par seconde avec vos clés KMS asymétriques RSA, plus 300 demandes supplémentaires par seconde avec vos clés KMS ECC.

Demandes d'API effectuées en votre nom

Vous pouvez effectuer des demandes d'API directement ou en utilisant un AWS service intégré qui envoie des demandes d'API AWS KMS en votre nom. Le quota s'applique aux deux types de demandes.

Par exemple, vous pouvez stocker des données dans Simple Storage Service (Amazon S3) à l'aide du chiffrement côté serveur avec une clé KMS (SSE-KMS). Chaque fois que vous chargez ou téléchargez un objet S3 chiffré avec SSE-KMS, Amazon S3 envoie une demande

`GenerateDataKey` (pour les chargements) ou `Decrypt` (pour les téléchargements) en votre nom AWS KMS . Ces demandes sont prises en compte dans votre quota. Par AWS KMS conséquent, elles sont limitées si vous dépassez un total combiné de 5 500 (ou 10 000 ou 50 000 selon vos Région AWS) chargements ou téléchargements par seconde d'objets S3 chiffrés avec SSE-KMS.

Demandes entre comptes

Lorsqu'une application Compte AWS utilise une clé KMS appartenant à un autre compte, on parle de demande entre comptes. Pour les demandes inter-comptes, AWS KMS limite le compte qui effectue les demandes, et non pas le compte qui possède la clé KMS. Par exemple, si une application du compte A utilise une clé KMS du compte B, l'utilisation de la clé KMS est uniquement soumise aux quotas du compte A.

Quotas de demandes de magasin de clés personnalisé

AWS KMS gère les quotas de demandes pour les opérations cryptographiques sur les clés KMS dans un magasin de clés personnalisé. Ces quotas de demandes sont calculés séparément pour chaque magasin de clés personnalisé.

Quota de requêtes de magasin de clés personnalisé	Valeur par défaut (demandes par seconde) pour chaque magasin de clés personnalisé	Ajustable
AWS CloudHSM quota de demandes de stockage de clés	1800	Non
Quota de demandes de magasin de clés externes	1800	Oui

> **Note**
>
> AWS KMS les quotas de demandes de stockage de clés personnalisés n'apparaissent pas dans la console Service Quotas. Vous ne pouvez ni consulter, ni gérer ces quotas à l'aide des opérations de l'API Service Quotas. Pour solliciter une modification de votre quota de demandes de magasin de clés externes, accédez au Centre AWS Support et créez une demande.

Si le AWS CloudHSM cluster associé à un magasin de AWS CloudHSM clés traite de nombreuses commandes, y compris celles qui ne sont pas liées au magasin de clés personnalisé, vous pourriez obtenir un AWS KMS `ThrottlingException` lower-than-expected at. Dans ce cas, réduisez votre taux de demandes à AWS KMS, réduisez la charge non liée ou utilisez un AWS CloudHSM cluster dédié pour votre magasin de AWS CloudHSM clés.
AWS KMS signale la limitation des demandes de stockage de clés externes dans la `ExternalKeyStoreThrottle` CloudWatch métrique. Vous pouvez utiliser cette métrique pour visualiser les modèles de limitation, créer des alertes et ajuster votre quota de demandes pour le magasin de clés externes.

Une demande d'opération de chiffrement sur une clé KMS d'un magasin de clés personnalisé compte pour deux quotas :

- Quota de taux de demandes d'opérations de chiffrement (symétrique) (par compte)

  Les demandes d'opérations de chiffrement sur les clés KMS d'un magasin de clés personnalisé sont comptabilisées dans le quota `Cryptographic operations (symmetric) request rate` de chaque région Compte AWS . Par exemple, dans l'est des États-Unis (Virginie du Nord) (us-east-1), Compte AWS chacune peut recevoir jusqu'à 100 000 demandes par seconde sur des clés KMS de chiffrement symétriques, y compris des demandes utilisant une clé KMS dans un magasin de clés personnalisé.
- Quota de demandes de magasin de clés personnalisé (par magasin de clés personnalisé)

  Les demandes d'opérations de chiffrement sur les clés KMS d'un magasin de clés personnalisé sont également comptabilisées dans le `Custom key store request quota` de 1 800 opérations par seconde. Ces quotas sont calculés séparément pour chaque magasin de clés personnalisé. Elles peuvent inclure des demandes provenant de plusieurs Comptes AWS utilisateurs de clés KMS dans le magasin de clés personnalisé.

Par exemple, une opération de chiffrement sur une clé KMS dans un magasin de clés personnalisé (quel que soit le type) de la région USA Est (Virginie du Nord) (us-east-1) est prise en compte dans le quota au niveau `Cryptographic operations (symmetric) request rate` du compte (100 000 demandes par seconde) pour son compte et sa région, et dans le calcul de a (1 800 demandes par seconde) pour son magasin de clés personnalisé. `Custom key store request quota` Toutefois, une demande d'opération de gestion, telle que celle portant sur une clé KMS dans un

magasin de clés personnalisé PutKeyPolicy, ne s'applique qu'au quota au niveau du compte (15 demandes par seconde).

Limitation des demandes AWS KMS

Pour garantir AWS KMS des réponses rapides et fiables aux demandes d'API de tous les clients, il limite les demandes d'API qui dépassent certaines limites.

Le throttling se produit lorsque l'on AWS KMS rejette une demande qui pourrait autrement être valide et renvoie une `ThrottlingException` erreur comme la suivante.

```
You have exceeded the rate at which you may call KMS. Reduce the frequency of your
 calls.
(Service: AWSKMS; Status Code: 400; Error Code: ThrottlingException; Request ID: <ID>
```

AWS KMS limite les demandes pour les conditions suivantes.

- Le taux de demandes par seconde dépasse le quota de AWS KMS demandes pour un compte et une région.

  Par exemple, si les utilisateurs de votre compte soumettent 1 000 `DescribeKey` demandes par seconde, toutes AWS KMS les `DescribeKey` demandes suivantes sont limitées au cours de cette seconde.

  Pour répondre à la limitation, utilisez une stratégie d'interruption et de nouvelle tentative. Cette stratégie est mise en œuvre automatiquement pour les erreurs HTTP 400 dans certains cas AWS SDKs.
- Une salve ou un taux élevé soutenu de demandes de modification de l'état de la même clé KMS. Cette condition est souvent connue sous le nom de « touche de raccourci ».

  Par exemple, si une application de votre compte envoie une volée persistante `EnableKey` et `DisableKey` demande la même clé KMS, le nombre de demandes AWS KMS est limité. Cette limitation se produit même si les demandes ne dépassent pas la limite de request-per-second demandes pour les opérations `EnableKey` et`DisableKey`.

  Pour répondre à la limitation, ajustez votre la logique d'application, afin qu'elle ne fasse que les demandes requises ou qu'elle consolide les demandes de plusieurs fonctions.
- Les demandes d'opérations sur les clés KMS dans un magasin de AWS CloudHSM clés peuvent être limitées à un lower-than-expected rythme tel que le AWS CloudHSM cluster associé au

magasin de AWS CloudHSM clés traite de nombreuses commandes, y compris celles qui ne sont pas liées au magasin de AWS CloudHSM clés.

(AWS KMS ne limite plus les demandes d'opérations sur les clés KMS dans un magasin de AWS CloudHSM clés lorsqu'aucune session PKCS #11 n'est disponible pour le cluster. AWS CloudHSM Au lieu de cela, il lance un `KMSInternalException` et vous recommande de réessayer votre demande.)

Pour afficher les tendances de vos taux de demande, utilisez la console Service Quotas. Vous pouvez également créer une CloudWatch alarme Amazon qui vous avertit lorsque le taux de demandes atteint un certain pourcentage de la valeur du quota. Pour plus de détails, consultez Gérer vos taux de demandes AWS KMS d'API à l'aide de Service Quotas et d'Amazon CloudWatch dans le blog sur la AWS sécurité.

Tous les AWS KMS quotas sont ajustables, à l'exception du quota de ressources de rotation à la demande et du quota de demandes de stockage de AWS CloudHSM clés. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas. Pour demander une réduction de quota, pour modifier un quota qui ne figure pas dans la liste des Quotas de Service, ou pour modifier un quota dans un pays Région AWS où les Quotas de Service pour AWS KMS ne sont pas disponibles, rendez-vous AWS Support au Centre et créez un dossier.

> **Note**
>
> AWS KMS les quotas de demandes de stockage de clés personnalisés n'apparaissent pas dans la console Service Quotas. Vous ne pouvez ni consulter, ni gérer ces quotas à l'aide des opérations de l'API Service Quotas. Pour solliciter une modification de votre quota de demandes de magasin de clés externes, accédez au Centre AWS Support et créez une demande.

Exemples de code pour AWS KMS l'utilisation AWS SDKs

Les exemples de code suivants montrent comment utiliser AWS KMS un kit de développement AWS logiciel (SDK).

Les principes de base sont des exemples de code qui vous montrent comment effectuer les opérations essentielles au sein d'un service.

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés.

Les Scénarios sont des exemples de code qui vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions au sein d'un même service ou combinés à d'autres Services AWS.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Mise en route

Bonjour AWS Key Management Service

Les exemples de code suivants montrent comment démarrer avec AWS Key Management Service.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.kms.KmsAsyncClient;
import software.amazon.awssdk.services.kms.model.ListKeysRequest;
import software.amazon.awssdk.services.kms.paginators.ListKeysPublisher;
import java.util.concurrent.CompletableFuture;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloKMS {
    public static void main(String[] args) {
        listAllKeys();
    }

    public static void listAllKeys() {
        KmsAsyncClient kmsAsyncClient = KmsAsyncClient.builder()
            .build();
        ListKeysRequest listKeysRequest = ListKeysRequest.builder()
            .limit(15)
            .build();

        /*
         * The `subscribe` method is required when using paginator methods in the
 AWS SDK
         * because paginator methods return an instance of a `ListKeysPublisher`,
 which is
         * based on a reactive stream. This allows asynchronous retrieval of
 paginated
         * results as they become available. By subscribing to the stream, we can
 process
         * each page of results as they are emitted.
         */
        ListKeysPublisher keysPublisher =
 kmsAsyncClient.listKeysPaginator(listKeysRequest);
        CompletableFuture<Void> future = keysPublisher
            .subscribe(r -> r.keys().forEach(key ->
                System.out.println("The key ARN is: " + key.keyArn() + ". The key
 Id is: " + key.keyId())))
            .whenComplete((result, exception) -> {
                if (exception != null) {
                    System.err.println("Error occurred: " +
 exception.getMessage());
                } else {
                    System.out.println("Successfully listed all keys.");
```

```
                }
            });

        try {
            future.join();
        } catch (Exception e) {
            System.err.println("Failed to list keys: " + e.getMessage());
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK for Java 2.x API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
include "vendor/autoload.php";

use Aws\Kms\KmsClient;

echo "This file shows how to connect to the KmsClient, uses a paginator to get
 the keys for the account, and lists the KeyIds for up to 10 keys.\n";

$client = new KmsClient([]);

$pageLength = 10; // Change this value to change the number of records shown, or
 to break up the result into pages.

$keys = [];
$keysPaginator = $client->getPaginator("ListKeys", ['Limit' => $pageLength]);
foreach($keysPaginator as $page){
    foreach($page['Keys'] as $index => $key){
```

```
            echo "The $index index Key's ID is: {$key['KeyId']}\n";
        }
        echo "End of page one of results. Alter the \$pageLength variable to see more
 results.\n";
        break;
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK pour PHP API.

Exemples de code

- Exemples de base pour AWS KMS l'utilisation AWS SDKs
  - Bonjour AWS Key Management Service
  - Apprenez les bases AWS KMS d'un AWS SDK
  - Actions d' AWS KMS utilisation AWS SDKs
    - Utilisation CreateAlias avec un AWS SDK ou une CLI
    - Utilisation CreateGrant avec un AWS SDK ou une CLI
    - Utilisation CreateKey avec un AWS SDK ou une CLI
    - Utilisation Decrypt avec un AWS SDK ou une CLI
    - Utilisation DeleteAlias avec un AWS SDK ou une CLI
    - Utilisation DescribeKey avec un AWS SDK ou une CLI
    - Utilisation DisableKey avec un AWS SDK ou une CLI
    - Utilisation EnableKey avec un AWS SDK ou une CLI
    - Utilisation EnableKeyRotation avec un AWS SDK ou une CLI
    - Utilisation Encrypt avec un AWS SDK ou une CLI
    - Utilisation GenerateDataKey avec un AWS SDK ou une CLI
    - Utilisation GenerateDataKeyWithoutPlaintext avec un AWS SDK ou une CLI
    - Utilisation GenerateRandom avec un AWS SDK ou une CLI
    - Utilisation GetKeyPolicy avec un AWS SDK ou une CLI
    - Utilisation ListAliases avec un AWS SDK ou une CLI
    - Utilisation ListGrants avec un AWS SDK ou une CLI

Exemples de base pour AWS KMS l'utilisation AWS SDKs

Les exemples de code suivants montrent comment utiliser les principes de base de AWS Key Management Service with AWS SDKs.

Exemples

- Utilisation EnableKeyRotation avec un AWS SDK ou une CLI
- Utilisation Encrypt avec un AWS SDK ou une CLI
- Utilisation GenerateDataKey avec un AWS SDK ou une CLI
- Utilisation GenerateDataKeyWithoutPlaintext avec un AWS SDK ou une CLI
- Utilisation GenerateRandom avec un AWS SDK ou une CLI
- Utilisation GetKeyPolicy avec un AWS SDK ou une CLI
- Utilisation ListAliases avec un AWS SDK ou une CLI
- Utilisation ListGrants avec un AWS SDK ou une CLI
- Utilisation ListKeyPolicies avec un AWS SDK ou une CLI
- Utilisation ListKeys avec un AWS SDK ou une CLI
- Utilisation PutKeyPolicy avec un AWS SDK ou une CLI
- Utilisation ReEncrypt avec un AWS SDK ou une CLI
- Utilisation RetireGrant avec un AWS SDK ou une CLI
- Utilisation RevokeGrant avec un AWS SDK ou une CLI
- Utilisation ScheduleKeyDeletion avec un AWS SDK ou une CLI
- Utilisation Sign avec un AWS SDK ou une CLI
- Utilisation TagResource avec un AWS SDK ou une CLI
- Utilisation UpdateAlias avec un AWS SDK ou une CLI
- Utilisation Verify avec un AWS SDK ou une CLI

Bonjour AWS Key Management Service

Les exemples de code suivants montrent comment démarrer avec AWS Key Management Service.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.kms.KmsAsyncClient;
import software.amazon.awssdk.services.kms.model.ListKeysRequest;
import software.amazon.awssdk.services.kms.paginators.ListKeysPublisher;
import java.util.concurrent.CompletableFuture;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloKMS {
    public static void main(String[] args) {
        listAllKeys();
    }

    public static void listAllKeys() {
        KmsAsyncClient kmsAsyncClient = KmsAsyncClient.builder()
            .build();
        ListKeysRequest listKeysRequest = ListKeysRequest.builder()
            .limit(15)
            .build();

        /*
         * The `subscribe` method is required when using paginator methods in the
 AWS SDK
         * because paginator methods return an instance of a `ListKeysPublisher`,
 which is
         * based on a reactive stream. This allows asynchronous retrieval of
 paginated
         * results as they become available. By subscribing to the stream, we can
 process
         * each page of results as they are emitted.
         */
        ListKeysPublisher keysPublisher =
 kmsAsyncClient.listKeysPaginator(listKeysRequest);
        CompletableFuture<Void> future = keysPublisher
            .subscribe(r -> r.keys().forEach(key ->
                System.out.println("The key ARN is: " + key.keyArn() + ". The key
 Id is: " + key.keyId())))
```

```
            .whenComplete((result, exception) -> {
                if (exception != null) {
                    System.err.println("Error occurred: " +
 exception.getMessage());
                } else {
                    System.out.println("Successfully listed all keys.");
                }
            });

        try {
            future.join();
        } catch (Exception e) {
            System.err.println("Failed to list keys: " + e.getMessage());
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK for Java 2.x API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
include "vendor/autoload.php";

use Aws\Kms\KmsClient;

echo "This file shows how to connect to the KmsClient, uses a paginator to get
 the keys for the account, and lists the KeyIds for up to 10 keys.\n";

$client = new KmsClient([]);
```

```
$pageLength = 10; // Change this value to change the number of records shown, or
 to break up the result into pages.

$keys = [];
$keysPaginator = $client->getPaginator("ListKeys", ['Limit' => $pageLength]);
foreach($keysPaginator as $page){
    foreach($page['Keys'] as $index => $key){
        echo "The $index index Key's ID is: {$key['KeyId']}\n";
    }
    echo "End of page one of results. Alter the \$pageLength variable to see more
 results.\n";
    break;
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK pour PHP API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Apprenez les bases AWS KMS d'un AWS SDK

Les exemples de code suivants montrent comment :

- Créer une clé KMS.
- Répertoriez les clés KMS de votre compte et obtenez des informations les concernant.
- Activez et désactivez les clés KMS.
- Générez une clé de données symétrique qui peut être utilisée pour le chiffrement côté client.
- Générez une clé asymétrique utilisée pour signer numériquement les données.
- Clés de tag.
- Supprimez les clés KMS.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario à une invite de commande.

```
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.regions.Region;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.kms.model.AlreadyExistsException;
import software.amazon.awssdk.services.kms.model.DisabledException;
import software.amazon.awssdk.services.kms.model.EnableKeyRotationResponse;
import software.amazon.awssdk.services.kms.model.KmsException;
import software.amazon.awssdk.services.kms.model.NotFoundException;
import software.amazon.awssdk.services.kms.model.RevokeGrantResponse;
import java.util.List;
import java.util.Scanner;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.CompletionException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class KMSScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    private static String accountId = "";
```

```
   private static final Logger logger =
LoggerFactory.getLogger(KMSScenario.class);

   static KMSActions kmsActions = new KMSActions();

   static Scanner scanner = new Scanner(System.in);

   static String aliasName = "alias/dev-encryption-key";

   public static void main(String[] args) {
       final String usage = """
           Usage: <granteePrincipal>

           Where:
              granteePrincipal - The principal (user, service account, or group)
to whom the grant or permission is being given.
           """;

       if (args.length != 1) {
           logger.info(usage);
           return;
       }
       String granteePrincipal = args[0];
       String policyName = "default";

       accountId = kmsActions.getAccountId();
       String keyDesc = "Created by the AWS KMS API";

       logger.info(DASHES);
       logger.info("""
           Welcome to the AWS Key Management SDK Basics scenario.

           This program demonstrates how to interact with AWS Key Management
using the AWS SDK for Java (v2).
           The AWS Key Management Service (KMS) is a secure and highly available
service that allows you to create
           and manage AWS KMS keys and control their use across a wide range of
AWS services and applications.
           KMS provides a centralized and unified approach to managing
encryption keys, making it easier to meet your
           data protection and regulatory compliance requirements.

           This Basics scenario creates two key types:
```

```
            - A symmetric encryption key is used to encrypt and decrypt data.
            - An asymmetric key used to digitally sign data.

            Let's get started...
            """);
        waitForInputToContinue(scanner);

        try {
        // Run the methods that belong to this scenario.
        String targetKeyId = runScenario(granteePrincipal, keyDesc, policyName);
        requestDeleteResources(aliasName, targetKeyId);

        } catch (Throwable rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
    }

    private static String runScenario(String granteePrincipal, String keyDesc,
String policyName) throws Throwable {
        logger.info(DASHES);
        logger.info("1. Create a symmetric KMS key\n");
        logger.info("First, the program will creates a symmetric KMS key that you
can used to encrypt and decrypt data.");
        waitForInputToContinue(scanner);
        String targetKeyId;
        try {
            CompletableFuture<String> futureKeyId =
kmsActions.createKeyAsync(keyDesc);
            targetKeyId = futureKeyId.join();
            logger.info("A symmetric key was successfully created " +
targetKeyId);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
```

```
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("""
            2. Enable a KMS key

            By default, when the SDK creates an AWS key, it is enabled. The next
bit of code checks to
            determine if the key is enabled.
             """);
        waitForInputToContinue(scanner);
        boolean isEnabled;
        try {
            CompletableFuture<Boolean> futureIsKeyEnabled =
kmsActions.isKeyEnabledAsync(targetKeyId);
            isEnabled = futureIsKeyEnabled.join();
            logger.info("Is the key enabled? {}", isEnabled);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }

        if (!isEnabled)
            try {
                CompletableFuture<Void> future =
kmsActions.enableKeyAsync(targetKeyId);
                future.join();

            } catch (RuntimeException rt) {
                Throwable cause = rt.getCause();
                if (cause instanceof KmsException kmsEx) {
                    logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
```

```
                } else {
                    logger.info("An unexpected error occurred: " +
rt.getMessage());
                }
                throw cause;
            }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("3. Encrypt data using the symmetric KMS key");
        String plaintext = "Hello, AWS KMS!";
        logger.info("""
            One of the main uses of symmetric keys is to encrypt and decrypt
data.
            Next, the code encrypts the string {} with the SYMMETRIC_DEFAULT
encryption algorithm.
            """, plaintext);
        waitForInputToContinue(scanner);
        SdkBytes encryptedData;
        try {
            CompletableFuture<SdkBytes> future =
kmsActions.encryptDataAsync(targetKeyId, plaintext);
            encryptedData = future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof DisabledException kmsDisabledEx) {
                logger.info("KMS error occurred due to a disabled
key: Error message: {}, Error code {}", kmsDisabledEx.getMessage(),
kmsDisabledEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("4. Create an alias");
        logger.info("""

            The alias name should be prefixed with 'alias/'.
            The default, 'alias/dev-encryption-key'.
```

```
             """);
        waitForInputToContinue(scanner);

        try {
            CompletableFuture<Void> future =
kmsActions.createCustomAliasAsync(targetKeyId, aliasName);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof AlreadyExistsException kmsExistsEx) {
                if (kmsExistsEx.getMessage().contains("already exists")) {
                    logger.info("The alias '" + aliasName + "' already exists.
Moving on...");
                }
            } else {
                logger.error("An unexpected error occurred: " + rt.getMessage(),
rt);
                deleteKey(targetKeyId);
                throw cause;
            }
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("5. List all of your aliases");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Object> future = kmsActions.listAllAliasesAsync();
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);
```

```
        logger.info(DASHES);
        logger.info("6. Enable automatic rotation of the KMS key");
        logger.info("""

            By default, when the SDK enables automatic rotation of a KMS key,
            KMS rotates the key material of the KMS key one year (approximately
365 days) from the enable date and every year
            thereafter.
            """);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<EnableKeyRotationResponse> future =
kmsActions.enableKeyRotationAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("""
            7. Create a grant

            A grant is a policy instrument that allows Amazon Web Services
principals to use KMS keys.
            It also can allow them to view a KMS key (DescribeKey) and create and
manage grants.
            When authorizing access to a KMS key, grants are considered along
with key policies and IAM policies.
            """);

        waitForInputToContinue(scanner);
        String grantId = null;
```

```
        try {
            CompletableFuture<String> futureGrantId =
kmsActions.grantKeyAsync(targetKeyId, granteePrincipal);
            grantId = futureGrantId.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("8. List grants for the KMS key");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Object> future =
kmsActions.displayGrantIdsAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("9. Revoke the grant");
        logger.info("""
```

```
            The revocation of a grant immediately removes the permissions and
access that the grant had provided.
            This means that any principal (user, role, or service) that was
granted access to perform specific
            KMS operations on a KMS key will no longer be able to perform those
operations.
            """);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<RevokeGrantResponse> future =
kmsActions.revokeKeyGrantAsync(targetKeyId, grantId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                if (kmsEx.getMessage().contains("Grant does not exist")) {
                    logger.info("The grant ID '" + grantId + "' does not exist.
Moving on...");
                } else {
                    logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
                    throw cause;
                }
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
                deleteAliasName(aliasName);
                deleteKey(targetKeyId);
                throw cause;
            }
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("10. Decrypt the data\n");
        logger.info("""
            Lets decrypt the data that was encrypted in an early step.
            The code uses the same key to decrypt the string that we encrypted
earlier in the program.
            """);
        waitForInputToContinue(scanner);
        String decryptedData = "";
        try {
```

```
            CompletableFuture<String> future =
kmsActions.decryptDataAsync(encryptedData, targetKeyId);
            decryptedData = future.join();
            logger.info("Decrypted data: " + decryptedData);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        logger.info("Decrypted text is: " + decryptedData);
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("11. Replace a key policy\n");
        logger.info("""
            A key policy is a resource policy for a KMS key. Key policies are the
primary way to control
            access to KMS keys. Every KMS key must have exactly one key policy.
The statements in the key policy
            determine who has permission to use the KMS key and how they can use
it.
            You can also use IAM policies and grants to control access to the KMS
key, but every KMS key
            must have a key policy.

            By default, when you create a key by using the SDK, a policy is
created that
            gives the AWS account that owns the KMS key full access to the KMS
key.

            Let's try to replace the automatically created policy with the
following policy.

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

```
                "Principal": {"AWS": "arn:aws:iam::0000000000:root"},
                "Action": "kms:*",
                "Resource": "*"
                }]
            """);

        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Boolean> future =
kmsActions.replacePolicyAsync(targetKeyId, policyName, accountId);
            boolean success = future.join();
            if (success) {
                logger.info("Key policy replacement succeeded.");
            } else {
                logger.error("Key policy replacement failed.");
            }

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("12. Get the key policy\n");
        logger.info("The next bit of code that runs gets the key policy to make
sure it exists.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<String> future =
kmsActions.getKeyPolicyAsync(targetKeyId, policyName);
            String policy = future.join();
            if (!policy.isEmpty()) {
                logger.info("Retrieved policy: " + policy);
            }
```

```
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("13. Create an asymmetric KMS key and sign your data\n");
        logger.info("""
             Signing your data with an AWS key can provide several benefits that
make it an attractive option
             for your data signing needs. By using an AWS KMS key, you can
leverage the
             security controls and compliance features provided by AWS,
             which can help you meet various regulatory requirements and enhance
the overall security posture
             of your organization.
            """);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Boolean> future = kmsActions.signVerifyDataAsync();
            boolean success = future.join();
            if (success) {
                logger.info("Sign and verify data operation succeeded.");
            } else {
                logger.error("Sign and verify data operation failed.");
            }

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
```

```
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("14. Tag your symmetric KMS Key\n");
        logger.info("""
            By using tags, you can improve the overall management, security, and
governance of your
            KMS keys, making it easier to organize, track, and control access to
your encrypted data within
            your AWS environment
            """);
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
kmsActions.tagKMSKeyAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            deleteAliasName(aliasName);
            deleteKey(targetKeyId);
            throw cause;
        }
        waitForInputToContinue(scanner);
        return targetKeyId;
    }

    // Deletes KMS resources with user input.
    private static void requestDeleteResources(String aliasName, String
targetKeyId) {
        logger.info(DASHES);
        logger.info("15. Schedule the deletion of the KMS key\n");
        logger.info("""
            By default, KMS applies a waiting period of 30 days,
```

```
            but you can specify a waiting period of 7-30 days. When this
 operation is successful,
            the key state of the KMS key changes to PendingDeletion and the key
 can't be used in any
            cryptographic operations. It remains in this state for the duration
 of the waiting period.

            Deleting a KMS key is a destructive and potentially dangerous
 operation. When a KMS key is deleted,
            all data that was encrypted under the KMS key is unrecoverable.
            """);
        logger.info("Would you like to delete the Key Management resources? (y/
n)");
        String delAns = scanner.nextLine().trim();
        if (delAns.equalsIgnoreCase("y")) {
            logger.info("You selected to delete the AWS KMS resources.");
            waitForInputToContinue(scanner);
            try {
                CompletableFuture<Void> future =
 kmsActions.deleteSpecificAliasAsync(aliasName);
                future.join();

            } catch (RuntimeException rt) {
                Throwable cause = rt.getCause();
                if (cause instanceof KmsException kmsEx) {
                    logger.info("KMS error occurred: Error message: {}, Error
 code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
                } else {
                    logger.info("An unexpected error occurred: " +
 rt.getMessage());
                }
            }
            waitForInputToContinue(scanner);
            try {
                CompletableFuture<Void> future =
 kmsActions.disableKeyAsync(targetKeyId);
                future.join();

            } catch (RuntimeException rt) {
                Throwable cause = rt.getCause();
                if (cause instanceof KmsException kmsEx) {
                    logger.info("KMS error occurred: Error message: {}, Error
 code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
                } else {
```

```
                    logger.info("An unexpected error occurred: " +
rt.getMessage());
                }
            }

            try {
                CompletableFuture<Void> future =
kmsActions.deleteKeyAsync(targetKeyId);
                future.join();

            } catch (RuntimeException rt) {
                Throwable cause = rt.getCause();
                if (cause instanceof KmsException kmsEx) {
                    logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
                } else {
                    logger.info("An unexpected error occurred: " +
rt.getMessage());
                }
            }

        } else {
            logger.info("The Key Management resources will not be deleted");
        }

        logger.info(DASHES);
        logger.info("This concludes the AWS Key Management SDK scenario");
        logger.info(DASHES);
    }

    // This method is invoked from Exceptions to clean up the resources.
    private static void deleteKey(String targetKeyId) {
        try {
            CompletableFuture<Void> future =
kmsActions.disableKeyAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
```

```
            }
        }

        try {
            CompletableFuture<Void> future =
kmsActions.deleteKeyAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
    }

    // This method is invoked from Exceptions to clean up the resources.
    private static void deleteAliasName(String aliasName) {
        try {
            CompletableFuture<Void> future =
kmsActions.deleteSpecificAliasAsync(aliasName);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
    }

    private static void waitForInputToContinue(Scanner scanner) {
        while (true) {
            logger.info("");
            logger.info("Enter 'c' followed by <ENTER> to continue:");
            String input = scanner.nextLine();

            if (input.trim().equalsIgnoreCase("c")) {
```

```
                logger.info("Continuing with the program...");
                logger.info("");
                break;
            } else {
                // Handle invalid input.
                logger.info("Invalid input. Please try again.");
            }
        }
    }
}
```

Définissez une classe qui englobe les actions KMS.

```
public class KMSActions {
    private static final Logger logger =
 LoggerFactory.getLogger(KMSActions.class);
    private static KmsAsyncClient kmsAsyncClient;

    /**
     * Retrieves an asynchronous AWS Key Management Service (KMS) client.
     * <p>
     * This method creates and returns a singleton instance of the KMS async
 client, with the following configurations:
     * <ul>
     *   <li>Max concurrency: 100</li>
     *   <li>Connection timeout: 60 seconds</li>
     *   <li>Read timeout: 60 seconds</li>
     *   <li>Write timeout: 60 seconds</li>
     *   <li>API call timeout: 2 minutes</li>
     *   <li>API call attempt timeout: 90 seconds</li>
     *   <li>Retry policy: up to 3 retries</li>
     *   <li>Credentials provider: environment variable credentials provider</li>
     * </ul>
     * <p>
     * If the client instance has already been created, it is returned instead of
 creating a new one.
     *
     * @return the KMS async client instance
     */
    private static KmsAsyncClient getAsyncClient() {
        if (kmsAsyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
```

```
                .maxConcurrency(100)
                .connectionTimeout(Duration.ofSeconds(60))
                .readTimeout(Duration.ofSeconds(60))
                .writeTimeout(Duration.ofSeconds(60))
                .build();

            ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
                .apiCallTimeout(Duration.ofMinutes(2))
                .apiCallAttemptTimeout(Duration.ofSeconds(90))
                .retryPolicy(RetryPolicy.builder()
                    .numRetries(3)
                    .build())
                .build();

            kmsAsyncClient = KmsAsyncClient.builder()
                .httpClient(httpClient)
                .overrideConfiguration(overrideConfig)
                .build();
        }
        return kmsAsyncClient;
    }

    /**
     * Creates a new symmetric encryption key asynchronously.
     *
     * @param keyDesc the description of the key to be created
     * @return a {@link CompletableFuture} that completes with the ID of the
newly created key
     * @throws RuntimeException if an error occurs while creating the key
     */
    public CompletableFuture<String> createKeyAsync(String keyDesc) {
        CreateKeyRequest keyRequest = CreateKeyRequest.builder()
            .description(keyDesc)
            .keySpec(KeySpec.SYMMETRIC_DEFAULT)
            .keyUsage(KeyUsageType.ENCRYPT_DECRYPT)
            .build();

        return getAsyncClient().createKey(keyRequest)
            .thenApply(resp -> resp.keyMetadata().keyId())
            .exceptionally(ex -> {
                throw new RuntimeException("An error occurred while creating the
key: " + ex.getMessage(), ex);
            });
```

```
    }

    /**
     * Asynchronously checks if a specified key is enabled.
     *
     * @param keyId the ID of the key to check
     * @return a {@link CompletableFuture} that, when completed, indicates
whether the key is enabled or not
     *
     * @throws RuntimeException if an exception occurs while checking the key
state
     */
    public CompletableFuture<Boolean> isKeyEnabledAsync(String keyId) {
        DescribeKeyRequest keyRequest = DescribeKeyRequest.builder()
            .keyId(keyId)
            .build();

        CompletableFuture<DescribeKeyResponse> responseFuture =
getAsyncClient().describeKey(keyRequest);
        return responseFuture.whenComplete((resp, ex) -> {
            if (resp != null) {
                KeyState keyState = resp.keyMetadata().keyState();
                if (keyState == KeyState.ENABLED) {
                    logger.info("The key is enabled.");
                } else {
                    logger.info("The key is not enabled. Key state: {}",
keyState);
                }
            } else {
                throw new RuntimeException(ex);
            }
        }).thenApply(resp -> resp.keyMetadata().keyState() == KeyState.ENABLED);
    }

    /**
     * Asynchronously enables the specified key.
     *
     * @param keyId the ID of the key to enable
     * @return a {@link CompletableFuture} that completes when the key has been
enabled
     */
    public CompletableFuture<Void> enableKeyAsync(String keyId) {
        EnableKeyRequest enableKeyRequest = EnableKeyRequest.builder()
            .keyId(keyId)
```

```
            .build();

        CompletableFuture<EnableKeyResponse> responseFuture =
getAsyncClient().enableKey(enableKeyRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
                logger.info("Key with ID [{}] has been enabled.", keyId);
            } else {
                if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("KMS error occurred while enabling
key: " + kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred
while enabling key: " + exception.getMessage(), exception);
                }
            }
        });

        return responseFuture.thenApply(response -> null);
    }

    /**
     * Encrypts the given text asynchronously using the specified KMS client and
key ID.
     *
     * @param keyId the ID of the KMS key to use for encryption
     * @param text the text to encrypt
     * @return a CompletableFuture that completes with the encrypted data as an
SdkBytes object
     */
    public CompletableFuture<SdkBytes> encryptDataAsync(String keyId, String
text) {
        SdkBytes myBytes = SdkBytes.fromUtf8String(text);
        EncryptRequest encryptRequest = EncryptRequest.builder()
            .keyId(keyId)
            .plaintext(myBytes)
            .build();

        CompletableFuture<EncryptResponse> responseFuture =
getAsyncClient().encrypt(encryptRequest).toCompletableFuture();
        return responseFuture.whenComplete((response, ex) -> {
            if (response != null) {
                String algorithm = response.encryptionAlgorithm().toString();
```

```
                logger.info("The string was encrypted with algorithm {}.",
algorithm);
            } else {
                throw new RuntimeException(ex);
            }
        }).thenApply(EncryptResponse::ciphertextBlob);
    }

    /**
     * Creates a custom alias for the specified target key asynchronously.
     *
     * @param targetKeyId the ID of the target key for the alias
     * @param aliasName   the name of the alias to create
     * @return a {@link CompletableFuture} that completes when the alias creation
operation is finished
     */
    public CompletableFuture<Void> createCustomAliasAsync(String targetKeyId,
String aliasName) {
        CreateAliasRequest aliasRequest = CreateAliasRequest.builder()
            .aliasName(aliasName)
            .targetKeyId(targetKeyId)
            .build();

        CompletableFuture<CreateAliasResponse> responseFuture =
getAsyncClient().createAlias(aliasRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
                logger.info("{} was successfully created.", aliasName);
            } else {
                if (exception instanceof ResourceExistsException) {
                    logger.info("Alias [{}] already exists. Moving on...",
aliasName);
                } else if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("KMS error occurred while creating
alias: " + kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred
while creating alias: " + exception.getMessage(), exception);
                }
            }
        });

        return responseFuture.thenApply(response -> null);
    }
```

```
   /**
    * Asynchronously lists all the aliases in the current AWS account.
    *
    * @return a {@link CompletableFuture} that completes when the list of
aliases has been processed
    */
   public CompletableFuture<Object> listAllAliasesAsync() {
       ListAliasesRequest aliasesRequest = ListAliasesRequest.builder()
           .limit(15)
           .build();

       ListAliasesPublisher paginator =
getAsyncClient().listAliasesPaginator(aliasesRequest);
       return paginator.subscribe(response -> {
               response.aliases().forEach(alias ->
                   logger.info("The alias name is: " + alias.aliasName())
               );
           })
           .thenApply(v -> null)
           .exceptionally(ex -> {
               if (ex.getCause() instanceof KmsException) {
                   KmsException e = (KmsException) ex.getCause();
                   throw new RuntimeException("A KMS exception occurred: " +
e.getMessage());
               } else {
                   throw new RuntimeException("An unexpected error occurred: " +
ex.getMessage());
               }
           });
   }

   /**
    * Enables key rotation asynchronously for the specified key ID.
    *
    * @param keyId the ID of the key for which to enable key rotation
    * @return a CompletableFuture that represents the asynchronous operation of
enabling key rotation
    * @throws RuntimeException if there was an error enabling key rotation,
either due to a KMS exception or an unexpected error
    */
   public CompletableFuture<EnableKeyRotationResponse>
enableKeyRotationAsync(String keyId) {
```

```
        EnableKeyRotationRequest enableKeyRotationRequest =
EnableKeyRotationRequest.builder()
            .keyId(keyId)
            .build();

        CompletableFuture<EnableKeyRotationResponse> responseFuture =
getAsyncClient().enableKeyRotation(enableKeyRotationRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
                logger.info("Key rotation has been enabled for key with id [{}]",
keyId);
            } else {
                if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("Failed to enable key rotation: "
+ kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
exception.getMessage(), exception);
                }
            }
        });

        return responseFuture;
    }

    /**
     * Grants permissions to a specified principal on a customer master key (CMK)
asynchronously.
     *
     * @param keyId             The unique identifier for the customer master key
(CMK) that the grant applies to.
     * @param granteePrincipal  The principal that is given permission to perform
the operations that the grant permits on the CMK.
     * @return A {@link CompletableFuture} that, when completed, contains the ID
of the created grant.
     * @throws RuntimeException If an error occurs during the grant creation
process.
     */
    public CompletableFuture<String> grantKeyAsync(String keyId, String
granteePrincipal) {
        List<GrantOperation> grantPermissions = List.of(
            GrantOperation.ENCRYPT,
            GrantOperation.DECRYPT,
            GrantOperation.DESCRIBE_KEY
```

```
        );

        CreateGrantRequest grantRequest = CreateGrantRequest.builder()
            .keyId(keyId)
            .name("grant1")
            .granteePrincipal(granteePrincipal)
            .operations(grantPermissions)
            .build();

        CompletableFuture<CreateGrantResponse> responseFuture =
getAsyncClient().createGrant(grantRequest);
        responseFuture.whenComplete((response, ex) -> {
            if (ex == null) {
                logger.info("Grant created successfully with ID: " +
response.grantId());
            } else {
                if (ex instanceof KmsException kmsEx) {
                    throw new RuntimeException("Failed to create grant: " +
kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
ex.getMessage(), ex);
                }
            }
        });

        return responseFuture.thenApply(CreateGrantResponse::grantId);
    }

    /**
     * Asynchronously displays the grant IDs for the specified key ID.
     *
     * @param keyId the ID of the AWS KMS key for which to list the grants
     * @return a {@link CompletableFuture} that, when completed, will be null
if the operation succeeded, or will throw a {@link RuntimeException} if the
operation failed
     * @throws RuntimeException if there was an error listing the grants, either
due to an {@link KmsException} or an unexpected error
     */
    public CompletableFuture<Object> displayGrantIdsAsync(String keyId) {
        ListGrantsRequest grantsRequest = ListGrantsRequest.builder()
            .keyId(keyId)
            .limit(15)
            .build();
```

```
        ListGrantsPublisher paginator =
getAsyncClient().listGrantsPaginator(grantsRequest);
        return paginator.subscribe(response -> {
                response.grants().forEach(grant -> {
                    logger.info("The grant Id is: " + grant.grantId());
                });
            })
            .thenApply(v -> null)
            .exceptionally(ex -> {
                Throwable cause = ex.getCause();
                if (cause instanceof KmsException) {
                    throw new RuntimeException("Failed to list grants: " +
cause.getMessage(), cause);
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
cause.getMessage(), cause);
                }
            });
    }

    /**
     * Revokes a grant for the specified AWS KMS key asynchronously.
     *
     * @param keyId   The ID or key ARN of the AWS KMS key.
     * @param grantId The identifier of the grant to be revoked.
     * @return A {@link CompletableFuture} representing the asynchronous
operation of revoking the grant.
     *         The {@link CompletableFuture} will complete with a {@link
RevokeGrantResponse} object
     *         if the operation is successful, or with a {@code null} value if an
error occurs.
     */
    public CompletableFuture<RevokeGrantResponse> revokeKeyGrantAsync(String
keyId, String grantId) {
        RevokeGrantRequest grantRequest = RevokeGrantRequest.builder()
            .keyId(keyId)
            .grantId(grantId)
            .build();

        CompletableFuture<RevokeGrantResponse> responseFuture =
getAsyncClient().revokeGrant(grantRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
```

```
                logger.info("Grant ID: [" + grantId + "] was successfully
revoked!");
            } else {
                if (exception instanceof KmsException kmsEx) {
                    if (kmsEx.getMessage().contains("Grant does not exist")) {
                        logger.info("The grant ID '" + grantId + "' does not
exist. Moving on...");
                    } else {
                        throw new RuntimeException("KMS error occurred: " +
kmsEx.getMessage(), kmsEx);
                    }
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
exception.getMessage(), exception);
                }
            }
        });

        return responseFuture;
    }


    /**
     * Asynchronously decrypts the given encrypted data using the specified key
ID.
     *
     * @param encryptedData The encrypted data to be decrypted.
     * @param keyId The ID of the key to be used for decryption.
     * @return A CompletableFuture that, when completed, will contain the
decrypted data as a String.
     *         If an error occurs during the decryption process, the
CompletableFuture will complete
     *         exceptionally with the error, and the method will return an empty
String.
     */
    public CompletableFuture<String> decryptDataAsync(SdkBytes encryptedData,
String keyId) {
        DecryptRequest decryptRequest = DecryptRequest.builder()
            .ciphertextBlob(encryptedData)
            .keyId(keyId)
            .build();

        CompletableFuture<DecryptResponse> responseFuture =
getAsyncClient().decrypt(decryptRequest);
```

```
        responseFuture.whenComplete((decryptResponse, exception) -> {
            if (exception == null) {
                logger.info("Data decrypted successfully for key ID: " + keyId);
            } else {
                if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("KMS error occurred while
decrypting data: " + kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred
while decrypting data: " + exception.getMessage(), exception);
                }
            }
        });

        return responseFuture.thenApply(decryptResponse ->
decryptResponse.plaintext().asString(StandardCharsets.UTF_8));
    }

    /**
     * Asynchronously replaces the policy for the specified KMS key.
     *
     * @param keyId       the ID of the KMS key to replace the policy for
     * @param policyName  the name of the policy to be replaced
     * @param accountId   the AWS account ID to be used in the policy
     * @return a {@link CompletableFuture} that completes with a boolean
indicating
     *         whether the policy replacement was successful or not
     */
    public CompletableFuture<Boolean> replacePolicyAsync(String keyId, String
policyName, String accountId) {
        String policy = """
    {
      "Version": "2012-10-17",
      "Statement": [{
        "Effect": "Allow",
        "Principal": {"AWS": "arn:aws:iam::%s:root"},
        "Action": "kms:*",
        "Resource": "*"
      }]
    }
    """.formatted(accountId);

        PutKeyPolicyRequest keyPolicyRequest = PutKeyPolicyRequest.builder()
            .keyId(keyId)
```

```
            .policyName(policyName)
            .policy(policy)
            .build();

        // First, get the current policy to check if it exists
        return getAsyncClient().getKeyPolicy(r ->
r.keyId(keyId).policyName(policyName))
            .thenCompose(response -> {
                logger.info("Current policy exists. Replacing it...");
                return getAsyncClient().putKeyPolicy(keyPolicyRequest);
            })
            .thenApply(putPolicyResponse -> {
                logger.info("The key policy has been replaced.");
                return true;
            })
            .exceptionally(throwable -> {
                if (throwable.getCause() instanceof LimitExceededException) {
                    logger.error("Cannot replace policy, as only one policy is
allowed per key.");
                    return false;
                }
                throw new RuntimeException("Error replacing policy", throwable);
            });
    }


    /**
     * Asynchronously retrieves the key policy for the specified key ID and
policy name.
     *
     * @param keyId       the ID of the AWS KMS key for which to retrieve the
policy
     * @param policyName the name of the key policy to retrieve
     * @return a {@link CompletableFuture} that, when completed, contains the key
policy as a {@link String}
     */
    public CompletableFuture<String> getKeyPolicyAsync(String keyId, String
policyName) {
        GetKeyPolicyRequest policyRequest = GetKeyPolicyRequest.builder()
            .keyId(keyId)
            .policyName(policyName)
            .build();

        return getAsyncClient().getKeyPolicy(policyRequest)
```

```
            .thenApply(response -> {
                String policy = response.policy();
                logger.info("The response is: " + policy);
                return policy;
            })
            .exceptionally(ex -> {
                throw new RuntimeException("Failed to get key policy", ex);
            });
    }

    /**
     * Asynchronously signs and verifies data using AWS KMS.
     *
     * <p>The method performs the following steps:
     * <ol>
     *     <li>Creates an AWS KMS key with the specified key spec, key usage, and
origin.</li>
     *     <li>Signs the provided message using the created KMS key and the
RSASSA-PSS-SHA-256 algorithm.</li>
     *     <li>Verifies the signature of the message using the created KMS key
and the RSASSA-PSS-SHA-256 algorithm.</li>
     * </ol>
     *
     * @return a {@link CompletableFuture} that completes with the result of the
signature verification,
     *         {@code true} if the signature is valid, {@code false} otherwise.
     * @throws KmsException if any error occurs during the KMS operations.
     * @throws RuntimeException if an unexpected error occurs.
     */
    public CompletableFuture<Boolean> signVerifyDataAsync() {
        String signMessage = "Here is the message that will be digitally signed";

        // Create an AWS KMS key used to digitally sign data.
        CreateKeyRequest createKeyRequest = CreateKeyRequest.builder()
            .keySpec(KeySpec.RSA_2048)
            .keyUsage(KeyUsageType.SIGN_VERIFY)
            .origin(OriginType.AWS_KMS)
            .build();

        return getAsyncClient().createKey(createKeyRequest)
            .thenCompose(createKeyResponse -> {
                String keyId = createKeyResponse.keyMetadata().keyId();
```

```
                SdkBytes messageBytes = SdkBytes.fromString(signMessage,
Charset.defaultCharset());
                SignRequest signRequest = SignRequest.builder()
                    .keyId(keyId)
                    .message(messageBytes)
                    .signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                    .build();

                return getAsyncClient().sign(signRequest)
                    .thenCompose(signResponse -> {
                        byte[] signedBytes =
signResponse.signature().asByteArray();

                        VerifyRequest verifyRequest = VerifyRequest.builder()
                            .keyId(keyId)

.message(SdkBytes.fromByteArray(signMessage.getBytes(Charset.defaultCharset())))

.signature(SdkBytes.fromByteBuffer(ByteBuffer.wrap(signedBytes)))

.signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                            .build();

                        return getAsyncClient().verify(verifyRequest)
                            .thenApply(verifyResponse -> {
                                return (boolean) verifyResponse.signatureValid();
                            });
                    });
            })
            .exceptionally(throwable -> {
               throw new RuntimeException("Failed to sign or verify data",
throwable);
            });
    }

    /**
     * Asynchronously tags a KMS key with a specific tag.
     *
     * @param keyId the ID of the KMS key to be tagged
     * @return a {@link CompletableFuture} that completes when the tagging
operation is finished
     */
    public CompletableFuture<Void> tagKMSKeyAsync(String keyId) {
        Tag tag = Tag.builder()
```

```
            .tagKey("Environment")
            .tagValue("Production")
            .build();

        TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
            .keyId(keyId)
            .tags(tag)
            .build();

        return getAsyncClient().tagResource(tagResourceRequest)
            .thenRun(() -> {
                logger.info("{} key was tagged", keyId);
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to tag the KMS key",
throwable);
            });
    }

    /**
     * Deletes a specific KMS alias asynchronously.
     *
     * @param aliasName the name of the alias to be deleted
     * @return a {@link CompletableFuture} representing the asynchronous
operation of deleting the specified alias
     */
    public CompletableFuture<Void> deleteSpecificAliasAsync(String aliasName) {
        DeleteAliasRequest deleteAliasRequest = DeleteAliasRequest.builder()
            .aliasName(aliasName)
            .build();

        return getAsyncClient().deleteAlias(deleteAliasRequest)
            .thenRun(() -> {
                logger.info("Alias {} has been deleted successfully", aliasName);
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to delete alias: " +
aliasName, throwable);
            });
    }

    /**
     * Asynchronously disables the specified AWS Key Management Service (KMS)
key.
```

```
    *
    * @param keyId the ID or Amazon Resource Name (ARN) of the KMS key to be
disabled
    * @return a CompletableFuture that, when completed, indicates that the key
has been disabled successfully
    */
   public CompletableFuture<Void> disableKeyAsync(String keyId) {
       DisableKeyRequest keyRequest = DisableKeyRequest.builder()
           .keyId(keyId)
           .build();

       return getAsyncClient().disableKey(keyRequest)
           .thenRun(() -> {
               logger.info("Key {} has been disabled successfully",keyId);
           })
           .exceptionally(throwable -> {
               throw new RuntimeException("Failed to disable key: " + keyId,
throwable);
           });
   }

   /**
    * Deletes a KMS key asynchronously.
    *
    * <p><strong>Warning:</strong> Deleting a KMS key is a destructive and
potentially dangerous operation.
    * When a KMS key is deleted, all data that was encrypted under the KMS key
becomes unrecoverable.
    * This means that any files, databases, or other data that were encrypted
using the deleted KMS key
    * will become permanently inaccessible. Exercise extreme caution when
deleting KMS keys.</p>
    *
    * @param keyId the ID of the KMS key to delete
    * @return a {@link CompletableFuture} that completes when the key deletion
is scheduled
    */
   public CompletableFuture<Void> deleteKeyAsync(String keyId) {
       ScheduleKeyDeletionRequest deletionRequest =
ScheduleKeyDeletionRequest.builder()
           .keyId(keyId)
           .pendingWindowInDays(7)
           .build();
```

```
        return getAsyncClient().scheduleKeyDeletion(deletionRequest)
            .thenRun(() -> {
                logger.info("Key {} will be deleted in 7 days", keyId);
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to schedule key deletion for
 key ID: " + keyId, throwable);
            });
    }


    public String getAccountId(){
        try (StsClient stsClient = StsClient.create()){
            GetCallerIdentityResponse callerIdentity =
 stsClient.getCallerIdentity();
            return callerIdentity.account();
        }
    }
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - CreateAlias
  - CreateGrant
  - CreateKey
  - Decrypt (Déchiffrer)
  - DescribeKey
  - DisableKey
  - EnableKey
  - Encrypt (Chiffrer)
  - GetKeyPolicy
  - ListAliases
  - ListGrants
  - ListKeys
  - RevokeGrant
  - ScheduleKeyDeletion

- Sign (Signer)
- TagResource

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        echo "\n";
        echo "-------------------------------------\n";
        echo <<<WELCOME
Welcome to the AWS Key Management Service SDK Basics scenario.

This program demonstrates how to interact with AWS Key Management Service using
 the AWS SDK for PHP (v3).
The AWS Key Management Service (KMS) is a secure and highly available service
 that allows you to create
and manage AWS KMS keys and control their use across a wide range of AWS services
 and applications.
KMS provides a centralized and unified approach to managing encryption keys,
 making it easier to meet your
data protection and regulatory compliance requirements.

This KMS Basics scenario creates two key types:
- A symmetric encryption key is used to encrypt and decrypt data.
- An asymmetric key used to digitally sign data.

Let's get started...\n
WELCOME;
        echo "-------------------------------------\n";
        $this->pressEnter();

        $this->kmsClient = new KmsClient([]);
        // Initialize the KmsService class with the client. This allows you to
 override any defaults in the client before giving it to the service class.
```

```
        $this->kmsService = new KmsService($this->kmsClient);

        // 1. Create a symmetric KMS key.
        echo "\n";
        echo "1. Create a symmetric KMS key.\n";
        echo "First, we will create a symmetric KMS key that is used to encrypt
 and decrypt data by invoking createKey().\n";
        $this->pressEnter();

        $key = $this->kmsService->createKey();
        $this->resources['symmetricKey'] = $key['KeyId'];
        echo "Created a customer key with ARN {$key['Arn']}.\n";
        $this->pressEnter();

        // 2. Enable a KMS key.
        echo "\n";
        echo "2. Enable a KMS key.\n";
        echo "By default when you create an AWS key, it is enabled. The code
 checks to
determine if the key is enabled. If it is not enabled, the code enables it.\n";
        $this->pressEnter();

        $keyInfo = $this->kmsService->describeKey($key['KeyId']);
        if(!$keyInfo['Enabled']){
            echo "The key was not enabled, so we will enable it.\n";
            $this->pressEnter();
            $this->kmsService->enableKey($key['KeyId']);
            echo "The key was successfully enabled.\n";
        }else{
            echo "The key was already enabled, so there was no need to enable it.
\n";
        }
        $this->pressEnter();

        // 3. Encrypt data using the symmetric KMS key.
        echo "\n";
        echo "3. Encrypt data using the symmetric KMS key.\n";
        echo "One of the main uses of symmetric keys is to encrypt and decrypt
 data.\n";
        echo "Next, we'll encrypt the string 'Hello, AWS KMS!' with the
 SYMMETRIC_DEFAULT encryption algorithm.\n";
        $this->pressEnter();
        $text = "Hello, AWS KMS!";
        $encryption = $this->kmsService->encrypt($key['KeyId'], $text);
```

```
        echo "The plaintext data was successfully encrypted with the algorithm:
 {$encryption['EncryptionAlgorithm']}.\n";
        $this->pressEnter();

        // 4. Create an alias.
        echo "\n";
        echo "4. Create an alias.\n";
        $aliasInput = testable_readline("Please enter an alias prefixed with
 \"alias/\" or press enter to use a default value: ");
        if($aliasInput == ""){
            $aliasInput = "alias/dev-encryption-key";
        }
        $this->kmsService->createAlias($key['KeyId'], $aliasInput);
        $this->resources['alias'] = $aliasInput;
        echo "The alias \"$aliasInput\" was successfully created.\n";
        $this->pressEnter();

        // 5. List all of your aliases.
        $aliasPageSize = 10;
        echo "\n";
        echo "5. List all of your aliases, up to $aliasPageSize.\n";
        $this->pressEnter();
        $aliasPaginator = $this->kmsService->listAliases();
        foreach($aliasPaginator as $pages){
            foreach($pages['Aliases'] as $alias){
                echo $alias['AliasName'] . "\n";
            }
            break;
        }
        $this->pressEnter();

        // 6. Enable automatic rotation of the KMS key.
        echo "\n";
        echo "6. Enable automatic rotation of the KMS key.\n";
        echo "By default, when the SDK enables automatic rotation of a KMS key,
KMS rotates the key material of the KMS key one year (approximately 365 days)
 from the enable date and every year
thereafter.";
        $this->pressEnter();
        $this->kmsService->enableKeyRotation($key['KeyId']);
        echo "The key's rotation was successfully set for key:
 {$key['KeyId']}\n";
        $this->pressEnter();
```

```
        // 7. Create a grant.
        echo "7. Create a grant.\n";
        echo "\n";
        echo "A grant is a policy instrument that allows Amazon Web Services
 principals to use KMS keys.
It also can allow them to view a KMS key (DescribeKey) and create and manage
 grants.
When authorizing access to a KMS key, grants are considered along with key
 policies and IAM policies.\n";
        $granteeARN = testable_readline("Please enter the Amazon Resource Name
 (ARN) of an Amazon Web Services principal. Valid principals include Amazon
 Web Services accounts, IAM users, IAM roles, federated users, and assumed
 role users. For help with the ARN syntax for a principal, see IAM ARNs in the
 Identity and Access Management User Guide. \nTo skip this step, press enter
 without any other values: ");
        if($granteeARN){
            $operations = [
                "ENCRYPT",
                "DECRYPT",
                "DESCRIBE_KEY",
            ];
            $grant = $this->kmsService->createGrant($key['KeyId'], $granteeARN,
 $operations);
            echo "The grant Id is: {$grant['GrantId']}\n";
        }else{
            echo "Steps 7, 8, and 9 will be skipped.\n";
        }
        $this->pressEnter();

        // 8. List grants for the KMS key.
        if($granteeARN){
            echo "8. List grants for the KMS key.\n\n";
            $grantsPaginator = $this->kmsService->listGrants($key['KeyId']);
            foreach($grantsPaginator as $page){
                foreach($page['Grants'] as $grant){
                    echo $grant['GrantId'] . "\n";
                }
            }
        }else{
            echo "Skipping step 8...\n";
        }
        $this->pressEnter();

        // 9. Revoke the grant.
```

```
        if($granteeARN) {
            echo "\n";
            echo "9. Revoke the grant.\n";
            $this->pressEnter();
            $this->kmsService->revokeGrant($grant['GrantId'], $keyInfo['KeyId']);
            echo "{$grant['GrantId']} was successfully revoked!\n";
        }else{
            echo "Skipping step 9...\n";
        }
        $this->pressEnter();

        // 10. Decrypt the data.
        echo "\n";
        echo "10. Decrypt the data.\n";
        echo "Let's decrypt the data that was encrypted before.\n";
        echo "We'll use the same key to decrypt the string that we encrypted
 earlier in the program.\n";
        $this->pressEnter();
        $decryption = $this->kmsService->decrypt($keyInfo['KeyId'],
 $encryption['CiphertextBlob'], $encryption['EncryptionAlgorithm']);
        echo "The decrypted text is: {$decryption['Plaintext']}\n";
        $this->pressEnter();

        // 11. Replace a Key Policy.
        echo "\n";
        echo "11. Replace a Key Policy.\n";
        echo "A key policy is a resource policy for a KMS key. Key policies are
 the primary way to control access to KMS keys.\n";
        echo "Every KMS key must have exactly one key policy. The statements in
 the key policy determine who has permission to use the KMS key and how they can
 use it.\n";
        echo " You can also use IAM policies and grants to control access to the
 KMS key, but every KMS key must have a key policy.\n";
        echo "We will replace the key's policy with a new one:\n";
        $stsClient = new StsClient([]);
        $result = $stsClient->getCallerIdentity();
        $accountId = $result['Account'];
        $keyPolicy = <<< KEYPOLICY
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {"AWS": "arn:aws:iam::$accountId:root"},
        "Action": "kms:*",
```

```
        "Resource": "*"
    }]
}
KEYPOLICY;
        echo $keyPolicy;
        $this->pressEnter();
        $this->kmsService->putKeyPolicy($keyInfo['KeyId'], $keyPolicy);
        echo "The Key Policy was successfully replaced!\n";
        $this->pressEnter();

        // 12. Retrieve the key policy.
        echo "\n";
        echo "12. Retrieve the key policy.\n";
        echo "Let's get some information about the new policy and print it to the
 screen.\n";
        $this->pressEnter();
        $policyInfo = $this->kmsService->getKeyPolicy($keyInfo['KeyId']);
        echo "We got the info! Here is the policy: \n";
        echo $policyInfo['Policy'] . "\n";
        $this->pressEnter();

        // 13. Create an asymmetric KMS key and sign data.
        echo "\n";
        echo "13. Create an asymmetric KMS key and sign data.\n";
        echo "Signing your data with an AWS key can provide several benefits that
 make it an attractive option for your data signing needs.\n";
        echo "By using an AWS KMS key, you can leverage the security controls and
 compliance features provided by AWS, which can help you meet various regulatory
 requirements and enhance the overall security posture of your organization.\n";
        echo "First we'll create the asymmetric key.\n";
        $this->pressEnter();
        $keySpec = "RSA_2048";
        $keyUsage = "SIGN_VERIFY";
        $asymmetricKey = $this->kmsService->createKey($keySpec, $keyUsage);
        $this->resources['asymmetricKey'] = $asymmetricKey['KeyId'];
        echo "Created the key with ID: {$asymmetricKey['KeyId']}\n";
        echo "Next, we'll sign the data.\n";
        $this->pressEnter();
        $algorithm = "RSASSA_PSS_SHA_256";
        $sign = $this->kmsService->sign($asymmetricKey['KeyId'], $text,
 $algorithm);
        $verify = $this->kmsService->verify($asymmetricKey['KeyId'], $text,
 $sign['Signature'], $algorithm);
        echo "Signature verification result: {$sign['signature']}\n";
```

```
        $this->pressEnter();

        // 14. Tag the symmetric KMS key.
        echo "\n";
        echo "14. Tag the symmetric KMS key.\n";
        echo "By using tags, you can improve the overall management, security,
 and governance of your KMS keys, making it easier to organize, track, and
 control access to your encrypted data within your AWS environment.\n";
        echo "Let's tag our symmetric key as Environment->Production\n";
        $this->pressEnter();
        $this->kmsService->tagResource($key['KeyId'], [
            [
                'TagKey' => "Environment",
                'TagValue' => "Production",
            ],
        ]);
        echo "The key was successfully tagged!\n";
        $this->pressEnter();

        // 15. Schedule the deletion of the KMS key
        echo "\n";
        echo "15. Schedule the deletion of the KMS key.\n";
        echo "By default, KMS applies a waiting period of 30 days, but you can
 specify a waiting period of 7-30 days.\n";
        echo "When this operation is successful, the key state of the KMS key
 changes to PendingDeletion and the key can't be used in any cryptographic
 operations.\n";
        echo "It remains in this state for the duration of the waiting period.\n
\n";

        echo "Deleting a KMS key is a destructive and potentially dangerous
 operation. When a KMS key is deleted, all data that was encrypted under the KMS
 key is unrecoverable.\n\n";

        $cleanUp = testable_readline("Would you like to delete the resources
 created during this scenario, including the keys? (y/n): ");
        if($cleanUp == "Y" || $cleanUp == "y"){
            $this->cleanUp();
        }

        echo
 "--------------------------------------------------------------------------------
\n";
        echo "This concludes the AWS Key Management SDK Basics scenario\n";
```

```
        echo
 "------------------------------------------------------------------------------
\n";



namespace Kms;

use Aws\Kms\Exception\KmsException;
use Aws\Kms\KmsClient;
use Aws\Result;
use Aws\ResultPaginator;
use AwsUtilities\AWSServiceClass;

class KmsService extends AWSServiceClass
{

    protected KmsClient $client;
    protected bool $verbose;

    /***
     * @param KmsClient|null $client
     * @param bool $verbose
     */
    public function __construct(KmsClient $client = null, bool $verbose = false)
    {
        $this->verbose = $verbose;
        if($client){
            $this->client = $client;
            return;
        }
        $this->client = new KmsClient([]);
    }


    /***
     * @param string $keySpec
     * @param string $keyUsage
     * @param string $description
     * @return array
     */
    public function createKey(string $keySpec = "", string $keyUsage = "", string
 $description = "Created by the SDK for PHP")
    {
```

```
        $parameters = ['Description' => $description];
        if($keySpec && $keyUsage){
            $parameters['KeySpec'] = $keySpec;
            $parameters['KeyUsage'] = $keyUsage;
        }
        try {
            $result = $this->client->createKey($parameters);
            return $result['KeyMetadata'];
        }catch(KmsException $caught){
            // Check for error specific to createKey operations
            if ($caught->getAwsErrorMessage() == "LimitExceededException"){
                echo "The request was rejected because a quota was exceeded. For
 more information, see Quotas in the Key Management Service Developer Guide.";
            }
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param string $ciphertext
     * @param string $algorithm
     * @return Result
     */
    public function decrypt(string $keyId, string $ciphertext, string $algorithm
 = "SYMMETRIC_DEFAULT")
    {
        try{
            return $this->client->decrypt([
                'CiphertextBlob' => $ciphertext,
                'EncryptionAlgorithm' => $algorithm,
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem decrypting the data: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

```
    /***
     * @param string $keyId
     * @param string $text
     * @return Result
     */
    public function encrypt(string $keyId, string $text)
    {
        try {
            return $this->client->encrypt([
                'KeyId' => $keyId,
                'Plaintext' => $text,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "DisabledException"){
                echo "The request was rejected because the specified KMS key is
not enabled.\n";
            }
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param int $limit
     * @return ResultPaginator
     */
    public function listAliases(string $keyId = "", int $limit = 0)
    {
        $args = [];
        if($keyId){
            $args['KeyId'] = $keyId;
        }
        if($limit){
            $args['Limit'] = $limit;
        }
        try{
            return $this->client->getPaginator("ListAliases", $args);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidMarkerException"){
                echo "The request was rejected because the marker that specifies
where pagination should next begin is not valid.\n";
            }
```

```
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param string $alias
     * @return void
     */
    public function createAlias(string $keyId, string $alias)
    {
        try{
            $this->client->createAlias([
                'TargetKeyId' => $keyId,
                'AliasName' => $alias,
            ]);
        }catch (KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidAliasNameException"){
                echo "The request was rejected because the specified alias name
is not valid.";
            }
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param string $granteePrincipal
     * @param array $operations
     * @param array $grantTokens
     * @return Result
     */
    public function createGrant(string $keyId, string $granteePrincipal, array
$operations, array $grantTokens = [])
    {
        $args = [
            'KeyId' => $keyId,
            'GranteePrincipal' => $granteePrincipal,
            'Operations' => $operations,
        ];
```

```
        if($grantTokens){
            $args['GrantTokens'] = $grantTokens;
        }
        try{
            return $this->client->createGrant($args);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidGrantTokenException"){
                echo "The request was rejected because the specified grant token
is not valid.\n";
            }
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @return array
     */
    public function describeKey(string $keyId)
    {
        try {
            $result = $this->client->describeKey([
                "KeyId" => $keyId,
            ]);
            return $result['KeyMetadata'];
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "The request was rejected because the specified entity or
resource could not be found.\n";
            }
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @return void
     */
    public function disableKey(string $keyId)
    {
```

```
        try {
            $this->client->disableKey([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem disabling the key: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @return void
     */
    public function enableKey(string $keyId)
    {
        try {
            $this->client->enableKey([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "The request was rejected because the specified entity or
 resource could not be found.\n";
            }
            throw $caught;
        }
    }



    /***
     * @return array
     */
    public function listKeys()
    {
        try {
            $contents = [];
            $paginator = $this->client->getPaginator("ListKeys");
            foreach($paginator as $result){
                foreach ($result['Content'] as $object) {
```

```
                    $contents[] = $object;
                }
            }
            return $contents;
        }catch(KmsException $caught){
            echo "There was a problem listing the keys: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @return Result
     */
    public function listGrants(string $keyId)
    {
        try{
            return $this->client->listGrants([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "    The request was rejected because the specified entity
 or resource could not be found.\n";
            }
            throw $caught;
        }
    }


    /***
     * @param string $keyId
     * @return Result
     */
    public function getKeyPolicy(string $keyId)
    {
        try {
            return $this->client->getKeyPolicy([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
```

```
            echo "There was a problem getting the key policy: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }


    /***
     * @param string $grantId
     * @param string $keyId
     * @return void
     */
    public function revokeGrant(string $grantId, string $keyId)
    {
        try{
            $this->client->revokeGrant([
                'GrantId' => $grantId,
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem with revoking the grant: {$caught-
>getAwsErrorMessage()}.\n";
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param int $pendingWindowInDays
     * @return void
     */
    public function scheduleKeyDeletion(string $keyId, int $pendingWindowInDays =
 7)
    {
        try {
            $this->client->scheduleKeyDeletion([
                'KeyId' => $keyId,
                'PendingWindowInDays' => $pendingWindowInDays,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem scheduling the key deletion: {$caught-
>getAwsErrorMessage()}\n";
```

```
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param array $tags
     * @return void
     */
    public function tagResource(string $keyId, array $tags)
    {
        try {
            $this->client->tagResource([
                'KeyId' => $keyId,
                'Tags' => $tags,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem applying the tag(s): {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param string $message
     * @param string $algorithm
     * @return Result
     */
    public function sign(string $keyId, string $message, string $algorithm)
    {
        try {
            return $this->client->sign([
                'KeyId' => $keyId,
                'Message' => $message,
                'SigningAlgorithm' => $algorithm,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem signing the data: {$caught-
>getAwsErrorMessage()}\n";
```

```
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param int $rotationPeriodInDays
     * @return void
     */
    public function enableKeyRotation(string $keyId, int $rotationPeriodInDays =
365)
    {
        try{
            $this->client->enableKeyRotation([
                'KeyId' => $keyId,
                'RotationPeriodInDays' => $rotationPeriodInDays,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "The request was rejected because the specified entity or
resource could not be found.\n";
            }
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param string $policy
     * @return void
     */
    public function putKeyPolicy(string $keyId, string $policy)
    {
        try {
            $this->client->putKeyPolicy([
                'KeyId' => $keyId,
                'Policy' => $policy,
            ]);
        }catch(KmsException $caught){
```

```
            echo "There was a problem replacing the key policy: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }



    /***
     * @param string $aliasName
     * @return void
     */
    public function deleteAlias(string $aliasName)
    {
        try {
            $this->client->deleteAlias([
                'AliasName' => $aliasName,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem deleting the alias: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }



    /***
     * @param string $keyId
     * @param string $message
     * @param string $signature
     * @param string $signingAlgorithm
     * @return bool
     */
    public function verify(string $keyId, string $message, string $signature,
 string $signingAlgorithm)
    {
        try {
            $result = $this->client->verify([
                'KeyId' => $keyId,
                'Message' => $message,
                'Signature' => $signature,
                'SigningAlgorithm' => $signingAlgorithm,
            ]);
```

```
            return $result['SignatureValid'];
        }catch(KmsException $caught){
            echo "There was a problem verifying the signature: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }


}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK pour PHP .
  - CreateAlias
  - CreateGrant
  - CreateKey
  - Decrypt (Déchiffrer)
  - DescribeKey
  - DisableKey
  - EnableKey
  - Encrypt (Chiffrer)
  - GetKeyPolicy
  - ListAliases
  - ListGrants
  - ListKeys
  - RevokeGrant
  - ScheduleKeyDeletion
  - Sign (Signer)
  - TagResource

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KMSScenario:
    """Runs an interactive scenario that shows how to get started with KMS."""

    def __init__(
        self,
        key_manager: KeyManager,
        key_encryption: KeyEncrypt,
        alias_manager: AliasManager,
        grant_manager: GrantManager,
        key_policy: KeyPolicy,
    ):
        self.key_manager = key_manager
        self.key_encryption = key_encryption
        self.alias_manager = alias_manager
        self.grant_manager = grant_manager
        self.key_policy = key_policy
        self.key_id = ""
        self.alias_name = ""
        self.asymmetric_key_id = ""

    def kms_scenario(self):
        key_description = "Created by the AWS KMS API"

        print(DASHES)
        print(
            """
Welcome to the AWS Key Management SDK Basics scenario.

This program demonstrates how to interact with AWS Key Management using the AWS
 SDK for Python (Boto3).
The AWS Key Management Service (KMS) is a secure and highly available service
 that allows you to create
```

```
and manage AWS KMS keys and control their use across a wide range of AWS services
 and applications.
KMS provides a centralized and unified approach to managing encryption keys,
 making it easier to meet your
data protection and regulatory compliance requirements.

This Basics scenario creates two key types:

- A symmetric encryption key is used to encrypt and decrypt data.
- An asymmetric key used to digitally sign data.

Let's get started...
        """
        )
        q.ask("Press Enter to continue...")

        print(DASHES)
        print(f"1. Create a symmetric KMS key\n")
        print(
            f"First, the program will creates a symmetric KMS key that you can
 used to encrypt and decrypt data."
        )
        q.ask("Press Enter to continue...")
        self.key_id = self.key_manager.create_key(key_description)["KeyId"]
        print(f"A symmetric key was successfully created {self.key_id}.")
        q.ask("Press Enter to continue...")
        print(DASHES)
        print(
            """
2. Enable a KMS key

By default, when the SDK creates an AWS key, it is enabled. The next bit of code
 checks to
determine if the key is enabled.
        """
        )
        q.ask("Press Enter to continue...")
        is_enabled = self.is_key_enabled(self.key_id)
        print(f"Is the key enabled? {is_enabled}")
        if not is_enabled:
            self.key_manager.enable_key(self.key_id)
        q.ask("Press Enter to continue...")
        print(DASHES)
        print(f"3. Encrypt data using the symmetric KMS key")
```

```
        plain_text = "Hello, AWS KMS!"
        print(
            f"""
One of the main uses of symmetric keys is to encrypt and decrypt data.
Next, the code encrypts the string "{plain_text}" with the SYMMETRIC_DEFAULT
 encryption algorithm.
        """
        )
        q.ask("Press Enter to continue...")
        encrypted_text = self.key_encryption.encrypt(self.key_id, plain_text)
        print(DASHES)
        print(f"4. Create an alias")
        print(
            """
Now, the program will create an alias for the KMS key. An alias is a friendly
 name that you
can associate with a KMS key. The alias name should be prefixed with 'alias/'.
        """
        )
        alias_name = q.ask("Enter an alias name: ", q.non_empty)
        self.alias_manager.create_alias(self.key_id, alias_name)
        print(f"{alias_name} was successfully created.")
        self.alias_name = alias_name
        print(DASHES)
        print(f"5. List all of your aliases")
        q.ask("Press Enter to continue...")
        self.alias_manager.list_aliases(10)
        q.ask("Press Enter to continue...")
        print(DASHES)
        print(f"6. Enable automatic rotation of the KMS key")
        print(
            """

By default, when the SDK enables automatic rotation of a KMS key,
KMS rotates the key material of the KMS key one year (approximately 365 days)
 from the enable date and every year
thereafter.
        """
        )
        q.ask("Press Enter to continue...")
        self.key_manager.enable_key_rotation(self.key_id)
        print(DASHES)
        print(f"Key rotation has been enabled for key with id {self.key_id}")
        print(
```

```
            """
7. Create a grant

A grant is a policy instrument that allows Amazon Web Services principals to use
 KMS keys.
It also can allow them to view a KMS key (DescribeKey) and create and manage
 grants.
When authorizing access to a KMS key, grants are considered along with key
 policies and IAM policies.
        """
        )
        print(
            """
To create a grant you must specify a account_id. To specify the grantee
 account_id, use the Amazon Resource Name (ARN)
of an AWS account_id. Valid principals include AWS accounts, IAM users, IAM
 roles, federated users,
and assumed role users.
        """
        )
        account_id = q.ask(
            "Enter an account_id, or press enter to skip creating a grant... "
        )
        grant = None
        if account_id != "":
            grant = self.grant_manager.create_grant(
                self.key_id,
                account_id,
                [
                    "Encrypt",
                    "Decrypt",
                    "DescribeKey",
                ],
            )
            print(f"Grant created successfully with ID: {grant['GrantId']}")

        q.ask("Press Enter to continue...")
        print(DASHES)
        print(DASHES)
        print(f"8. List grants for the KMS key")
        q.ask("Press Enter to continue...")
        self.grant_manager.list_grants(self.key_id)
        q.ask("Press Enter to continue...")
        print(DASHES)
```

```
        print(f"9. Revoke the grant")
        print(
            """
The revocation of a grant immediately removes the permissions and access that the
 grant had provided.
This means that any account_id (user, role, or service) that was granted access
 to perform specific
KMS operations on a KMS key will no longer be able to perform those operations.
        """
        )
        q.ask("Press Enter to continue...")

        if grant is not None:
            self.grant_manager.revoke_grant(self.key_id, grant["GrantId"])
            print(f"Grant ID: {grant['GrantId']} was successfully revoked!")

        q.ask("Press Enter to continue...")
        print(DASHES)
        print(f"10. Decrypt the data\n")
        print(
            """
Lets decrypt the data that was encrypted in an early step.
The code uses the same key to decrypt the string that we encrypted earlier in the
 program.
        """
        )
        q.ask("Press Enter to continue...")
        decrypted_data = self.key_encryption.decrypt(self.key_id, encrypted_text)
        print(f"Data decrypted successfully for key ID: {self.key_id}")
        print(f"Decrypted data: {decrypted_data}")

        q.ask("Press Enter to continue...")
        print(DASHES)
        print(f"11. Replace a key policy\n")
        print(
            """
A key policy is a resource policy for a KMS key. Key policies are the primary way
 to control
access to KMS keys. Every KMS key must have exactly one key policy. The
 statements in the key policy
determine who has permission to use the KMS key and how they can use it.
You can also use IAM policies and grants to control access to the KMS key, but
 every KMS key
must have a key policy.
```

```
By default, when you create a key by using the SDK, a policy is created that
gives the AWS account that owns the KMS key full access to the KMS key.

Let's try to replace the automatically created policy with the following policy.
{
"Version": "2012-10-17",
"Statement": [{
"Effect": "Allow",
"Principal": {"AWS": "arn:aws:iam::0000000000:root"},
"Action": "kms:*",
"Resource": "*"
}]
}
        """
        )
        account_id = q.ask("Enter your account ID or press enter to skip: ")
        if account_id != "":
            policy = {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": f"arn:aws:iam::{account_id}:root"},
                        "Action": "kms:*",
                        "Resource": "*",
                    }
                ],
            }

            self.key_policy.set_new_policy(self.key_id, policy)
            print("Key policy replacement succeeded.")
            q.ask("Press Enter to continue...")
        else:
            print("Skipping replacing the key policy.")

        print(DASHES)
        print(f"12. Get the key policy\n")
        print(
            f"The next bit of code that runs gets the key policy to make sure it
 exists."
        )
        q.ask("Press Enter to continue...")
        policy = self.key_policy.get_policy(self.key_id)
```

```
        print(f"The key policy is: {policy}")

        q.ask("Press Enter to continue...")
        print(DASHES)
        print(f"13. Create an asymmetric KMS key and sign your data\n")
        print(
            """
        Signing your data with an AWS key can provide several benefits that make
it an attractive option
        for your data signing needs. By using an AWS KMS key, you can leverage
the
        security controls and compliance features provided by AWS,
        which can help you meet various regulatory requirements and enhance the
overall security posture
        of your organization.
        """
        )
        q.ask("Press Enter to continue...")
        print(f"Sign and verify data operation succeeded.")
        self.asymmetric_key_id = self.key_manager.create_asymmetric_key()
        message = "Here is the message that will be digitally signed"
        signature = self.key_encryption.sign(self.asymmetric_key_id, message)
        if self.key_encryption.verify(self.asymmetric_key_id, message,
signature):
            print("Signature verification succeeded.")
        else:
            print("Signature verification failed.")

        q.ask("Press Enter to continue...")
        print(DASHES)
        print(f"14. Tag your symmetric KMS Key\n")
        print(
            """
        By using tags, you can improve the overall management, security, and
governance of your
        KMS keys, making it easier to organize, track, and control access to your
encrypted data within
        your AWS environment
        """
        )
        q.ask("Press Enter to continue...")
        self.key_manager.tag_resource(self.key_id, "Environment", "Production")
        self.clean_up()
```

```
    def is_key_enabled(self, key_id: str) -> bool:
        """
        Check if the key is enabled or not.

        :param key_id: The key to check.
        :return: True if the key is enabled, otherwise False.
        """
        response = self.key_manager.describe_key(key_id)
        return response["Enabled"] is True

    def clean_up(self):
        """
        Delete resources created by this scenario.
        """
        if self.alias_name != "":
            print(f"Deleting the alias {self.alias_name}.")
            self.alias_manager.delete_alias(self.alias_name)
        window = 7  # The window in days for a scheduled deletion.
        if self.key_id != "":
            print(
                """
Warning:
Deleting a KMS key is a destructive and potentially dangerous operation. When a
 KMS key is deleted,
all data that was encrypted under the KMS key is unrecoverable.
                """
            )
            if q.ask(
                f"Do you want to delete the key with ID {self.key_id} (y/n)?",
                q.is_yesno,
            ):
                print(
                    f"The key {self.key_id} will be deleted with a window of
 {window} days. You can cancel the deletion before"
                )
                print("the window expires.")
                self.key_manager.delete_key(self.key_id, window)
                self.key_id = ""

        if self.asymmetric_key_id != "":
            if q.ask(
                f"Do you want to delete the asymmetric key with ID
 {self.asymmetric_key_id} (y/n)?",
                q.is_yesno,
```

```
            ):
                print(
                    f"The key {self.asymmetric_key_id} will be deleted with a
 window of {window} days. You can cancel the deletion before"
                )
                print("the window expires.")
                self.key_manager.delete_key(self.asymmetric_key_id, window)
                self.asymmetric_key_id = ""


if __name__ == "__main__":
    kms_scenario = None
    try:
        kms_client = boto3.client("kms")
        a_key_manager = KeyManager(kms_client)
        a_key_encrypt = KeyEncrypt(kms_client)
        an_alias_manager = AliasManager(kms_client)
        a_grant_manager = GrantManager(kms_client)
        a_key_policy = KeyPolicy(kms_client)
        kms_scenario = KMSScenario(
            key_manager=a_key_manager,
            key_encryption=a_key_encrypt,
            alias_manager=an_alias_manager,
            grant_manager=a_grant_manager,
            key_policy=a_key_policy,
        )
        kms_scenario.kms_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo!")
        if kms_scenario is not None:
            kms_scenario.clean_up()
```

Classe Wrapper et méthodes pour la gestion des clés KMS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
```

```
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def create_key(self, key_description: str) -> dict[str, any]:
        """
        Creates a key with a user-provided description.

        :param key_description: A description for the key.
        :return: The key ID.
        """
        try:
            key = self.kms_client.create_key(Description=key_description)
["KeyMetadata"]
            self.created_keys.append(key)
            return key
        except ClientError as err:
            logging.error(
                "Couldn't create your key. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise


    def describe_key(self, key_id: str) -> dict[str, any]:
        """
        Describes a key.

        :param key_id: The ARN or ID of the key to describe.
        :return: Information about the key.
        """

        try:
            key = self.kms_client.describe_key(KeyId=key_id)["KeyMetadata"]
            return key
        except ClientError as err:
            logging.error(
                "Couldn't get key '%s'. Here's why: %s",
```

```
                key_id,
                err.response["Error"]["Message"],
            )
            raise


    def enable_key_rotation(self, key_id: str) -> None:
        """
        Enables rotation for a key.

        :param key_id: The ARN or ID of the key to enable rotation for.
        """
        try:
            self.kms_client.enable_key_rotation(KeyId=key_id)
        except ClientError as err:
            logging.error(
                "Couldn't enable rotation for key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise


    def create_asymmetric_key(self) -> str:
        """
        Creates an asymmetric key in AWS KMS for signing messages.

        :return: The ID of the created key.
        """
        try:
            key = self.kms_client.create_key(
                KeySpec="RSA_2048", KeyUsage="SIGN_VERIFY", Origin="AWS_KMS"
            )["KeyMetadata"]
            self.created_keys.append(key)
            return key["KeyId"]
        except ClientError as err:
            logger.error(
                "Couldn't create your key. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise


    def tag_resource(self, key_id: str, tag_key: str, tag_value: str) -> None:
```

```
        """
        Add or edit tags on a customer managed key.

        :param key_id: The ARN or ID of the key to enable rotation for.
        :param tag_key: Key for the tag.
        :param tag_value: Value for the tag.
        """
        try:
            self.kms_client.tag_resource(
                KeyId=key_id, Tags=[{"TagKey": tag_key, "TagValue": tag_value}]
            )
        except ClientError as err:
            logging.error(
                "Couldn't add a tag for the key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise


    def delete_key(self, key_id: str, window: int) -> None:
        """
        Deletes a list of keys.

        Warning:
        Deleting a KMS key is a destructive and potentially dangerous operation.
When a KMS key is deleted,
        all data that was encrypted under the KMS key is unrecoverable.

        :param key_id: The ARN or ID of the key to delete.
        :param window: The waiting period, in days, before the KMS key is
deleted.
        """

        try:
            self.kms_client.schedule_key_deletion(
                KeyId=key_id, PendingWindowInDays=window
            )
        except ClientError as err:
            logging.error(
                "Couldn't delete key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
```

```
            raise
```

Classe et méthodes Wrapper pour les alias de clé KMS.

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def create_alias(self, key_id: str, alias: str) -> None:
        """
        Creates an alias for the specified key.

        :param key_id: The ARN or ID of a key to give an alias.
        :param alias: The alias to assign to the key.
        """
        try:
            self.kms_client.create_alias(AliasName=alias, TargetKeyId=key_id)
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExistsException":
                logger.error(
                    "Could not create the alias %s because it already exists.",
 key_id
                )
            else:
                logger.error(
                    "Couldn't encrypt text. Here's why: %s",
                    err.response["Error"]["Message"],
                )
```

```
                raise


    def list_aliases(self, page_size: int) -> None:
        """
        Lists aliases for the current account.
        :param page_size: The number of aliases to list per page.
        """
        try:
            alias_paginator = self.kms_client.get_paginator("list_aliases")
            for alias_page in alias_paginator.paginate(
                PaginationConfig={"PageSize": page_size}
            ):
                print(f"Here are {page_size} aliases:")
                pprint(alias_page["Aliases"])
                if alias_page["Truncated"]:
                    answer = input(
                        f"Do you want to see the next {page_size} aliases (y/n)?
"
                    )
                    if answer.lower() != "y":
                        break
                else:
                    print("That's all your aliases!")
        except ClientError as err:
            logging.error(
                "Couldn't list your aliases. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise


    def delete_alias(self, alias: str) -> None:
        """
        Deletes an alias.

        :param alias: The alias to delete.
        """
        try:
            self.kms_client.delete_alias(AliasName=alias)
        except ClientError as err:
            logger.error(
                "Couldn't delete alias %s. Here's why: %s",
                alias,
```

```
                err.response["Error"]["Message"],
            )
            raise
```

Classe Wrapper et méthodes pour le chiffrement des clés KMS.

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def encrypt(self, key_id: str, text: str) -> str:
        """
        Encrypts text by using the specified key.

        :param key_id: The ARN or ID of the key to use for encryption.
        :param text: The text to encrypt.
        :return: The encrypted version of the text.
        """
        try:
            response = self.kms_client.encrypt(KeyId=key_id,
 Plaintext=text.encode())
            print(
                f"The string was encrypted with algorithm
 {response['EncryptionAlgorithm']}"
            )
            return response["CiphertextBlob"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DisabledException":
```

```
                logger.error(
                    "Could not encrypt because the key %s is disabled.", key_id
                )
            else:
                logger.error(
                    "Couldn't encrypt text. Here's why: %s",
                    err.response["Error"]["Message"],
                )
            raise


    def decrypt(self, key_id: str, cipher_text: str) -> bytes:
        """
        Decrypts text previously encrypted with a key.

        :param key_id: The ARN or ID of the key used to decrypt the data.
        :param cipher_text: The encrypted text to decrypt.
        :return: The decrypted text.
        """
        try:
            return self.kms_client.decrypt(KeyId=key_id,
CiphertextBlob=cipher_text)[
                "Plaintext"
            ]
        except ClientError as err:
            logger.error(
                "Couldn't decrypt your ciphertext. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise


    def sign(self, key_id: str, message: str) -> str:
        """
        Signs a message with a key.

        :param key_id: The ARN or ID of the key to use for signing.
        :param message: The message to sign.
        :return: The signature of the message.
        """
        try:
            return self.kms_client.sign(
                KeyId=key_id,
                Message=message.encode(),
```

```
                SigningAlgorithm="RSASSA_PSS_SHA_256",
            )["Signature"]
        except ClientError as err:
            logger.error(
                "Couldn't sign your message. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise


    def verify(self, key_id: str, message: str, signature: str) -> bool:
        """
        Verifies a signature against a message.

        :param key_id: The ARN or ID of the key used to sign the message.
        :param message: The message to verify.
        :param signature: The signature to verify.
        :return: True when the signature matches the message, otherwise False.
        """
        try:
            response = self.kms_client.verify(
                KeyId=key_id,
                Message=message.encode(),
                Signature=signature,
                SigningAlgorithm="RSASSA_PSS_SHA_256",
            )
            valid = response["SignatureValid"]
            print(f"The signature is {'valid' if valid else 'invalid'}.")
            return valid
        except ClientError as err:
            if err.response["Error"]["Code"] == "SignatureDoesNotMatchException":
                print("The signature is not valid.")
            else:
                logger.error(
                    "Couldn't verify your signature. Here's why: %s",
                    err.response["Error"]["Message"],
                )
                raise
```

Classe et méthodes Wrapper pour l'attribution de clés KMS.

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def create_grant(
        self, key_id: str, principal: str, operations: [str]
    ) -> dict[str, str]:
        """
        Creates a grant for a key that lets a principal generate a symmetric data
        encryption key.

        :param key_id: The ARN or ID of the key.
        :param principal: The principal to grant permission to.
        :param operations: The operations to grant permission for.
        :return: The grant that is created.
        """
        try:
            return self.kms_client.create_grant(
                KeyId=key_id,
                GranteePrincipal=principal,
                Operations=operations,
            )
        except ClientError as err:
            logger.error(
                "Couldn't create a grant on key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

```
    def list_grants(self, key_id):
        """
        Lists grants for a key.

        :param key_id: The ARN or ID of the key to query.
        :return: The grants for the key.
        """
        try:
            paginator = self.kms_client.get_paginator("list_grants")
            grants = []
            page_iterator = paginator.paginate(KeyId=key_id)
            for page in page_iterator:
                grants.extend(page["Grants"])

            print(f"Grants for key {key_id}:")
            pprint(grants)
            return grants
        except ClientError as err:
            logger.error(
                "Couldn't list grants for key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise


    def revoke_grant(self, key_id: str, grant_id: str) -> None:
        """
        Revokes a grant so that it can no longer be used.

        :param key_id: The ARN or ID of the key associated with the grant.
        :param grant_id: The ID of the grant to revoke.
        """
        try:
            self.kms_client.revoke_grant(KeyId=key_id, GrantId=grant_id)
        except ClientError as err:
            logger.error(
                "Couldn't revoke grant %s. Here's why: %s",
                grant_id,
                err.response["Error"]["Message"],
            )
            raise
```

Classe et méthodes Wrapper pour les politiques clés KMS.

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.

        :return: An instance of KeyPolicy initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def set_new_policy(self, key_id: str, policy: dict[str, any]) -> None:
        """
        Sets the policy of a key. Setting a policy entirely overwrites the
 existing
        policy, so care is taken to add a statement to the existing list of
 statements
        rather than simply writing a new policy.

        :param key_id: The ARN or ID of the key to set the policy to.
        :param policy: A new key policy. The key policy must allow the calling
 principal to make a subsequent
                       PutKeyPolicy request on the KMS key. This reduces the risk
 that the KMS key becomes unmanageable
        """

        try:
            self.kms_client.put_key_policy(KeyId=key_id,
 Policy=json.dumps(policy))
        except ClientError as err:
            logger.error(
                "Couldn't set policy for key %s. Here's why %s",
                key_id,
                err.response["Error"]["Message"],
```

```
            )
            raise



    def get_policy(self, key_id: str) -> dict[str, str]:
        """
        Gets the policy of a key.

        :param key_id: The ARN or ID of the key to query.
        :return: The key policy as a dict.
        """
        if key_id != "":
            try:
                response = self.kms_client.get_key_policy(
                    KeyId=key_id,
                )
                policy = json.loads(response["Policy"])
            except ClientError as err:
                logger.error(
                    "Couldn't get policy for key %s. Here's why: %s",
                    key_id,
                    err.response["Error"]["Message"],
                )
                raise
            else:
                pprint(policy)
                return policy
        else:
            print("Skipping get policy demo.")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - CreateAlias
  - CreateGrant
  - CreateKey
  - Decrypt (Déchiffrer)
  - DescribeKey
  - DisableKey

- EnableKey
- Encrypt (Chiffrer)
- GetKeyPolicy
- ListAliases
- ListGrants
- ListKeys
- RevokeGrant
- ScheduleKeyDeletion
- Sign (Signer)
- TagResource

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Actions d' AWS KMS utilisation AWS SDKs

Les exemples de code suivants montrent comment effectuer des AWS KMS actions individuelles avec AWS SDKs. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Ces extraits appellent l' AWS KMS API et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Vous pouvez voir les actions dans leur contexte dans Scénarios d' AWS KMS utilisation AWS SDKs .

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, veuillez consulter la AWS Key Management Service Référence d'API.

Exemples

- Utilisation CreateAlias avec un AWS SDK ou une CLI
- Utilisation CreateGrant avec un AWS SDK ou une CLI
- Utilisation CreateKey avec un AWS SDK ou une CLI
- Utilisation Decrypt avec un AWS SDK ou une CLI
- Utilisation DeleteAlias avec un AWS SDK ou une CLI

- Utilisation DescribeKey avec un AWS SDK ou une CLI
- Utilisation DisableKey avec un AWS SDK ou une CLI
- Utilisation EnableKey avec un AWS SDK ou une CLI
- Utilisation EnableKeyRotation avec un AWS SDK ou une CLI
- Utilisation Encrypt avec un AWS SDK ou une CLI
- Utilisation GenerateDataKey avec un AWS SDK ou une CLI
- Utilisation GenerateDataKeyWithoutPlaintext avec un AWS SDK ou une CLI
- Utilisation GenerateRandom avec un AWS SDK ou une CLI
- Utilisation GetKeyPolicy avec un AWS SDK ou une CLI
- Utilisation ListAliases avec un AWS SDK ou une CLI
- Utilisation ListGrants avec un AWS SDK ou une CLI
- Utilisation ListKeyPolicies avec un AWS SDK ou une CLI
- Utilisation ListKeys avec un AWS SDK ou une CLI
- Utilisation PutKeyPolicy avec un AWS SDK ou une CLI
- Utilisation ReEncrypt avec un AWS SDK ou une CLI
- Utilisation RetireGrant avec un AWS SDK ou une CLI
- Utilisation RevokeGrant avec un AWS SDK ou une CLI
- Utilisation ScheduleKeyDeletion avec un AWS SDK ou une CLI
- Utilisation Sign avec un AWS SDK ou une CLI
- Utilisation TagResource avec un AWS SDK ou une CLI
- Utilisation UpdateAlias avec un AWS SDK ou une CLI
- Utilisation Verify avec un AWS SDK ou une CLI

Utilisation **`CreateAlias`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `CreateAlias`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Creates an alias for an AWS Key Management Service (AWS KMS) key.
/// </summary>
public class CreateAlias
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();

        // The alias name must start with alias/ and can be
        // up to 256 alphanumeric characters long.
        var aliasName = "alias/ExampleAlias";

        // The value supplied as the TargetKeyId can be either
        // the key ID or key Amazon Resource Name (ARN) of the
        // AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";

        var request = new CreateAliasRequest
        {
            AliasName = aliasName,
            TargetKeyId = keyId,
        };

        var response = await client.CreateAliasAsync(request);
```

```
            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"Alias, {aliasName}, successfully created.");
            }
            else
            {
                Console.WriteLine($"Could not create alias.");
            }
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateAliasà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour créer un alias pour une clé KMS

La `create-alias` commande suivante crée un alias nommé `example-alias` pour la clé KMS identifiée par son ID de clé`1234abcd-12ab-34cd-56ef-1234567890ab`.

Les noms d'alias doivent commencer par`alias/`. N'utilisez pas de noms d'alias commençant par `alias/aws` ; ils sont réservés à l'usage de AWS.

```
aws kms create-alias \
    --alias-name alias/example-alias \
    --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette commande ne renvoie aucune sortie. Pour voir le nouvel alias, utilisez la `list-aliases` commande.

Pour plus d'informations, consultez la section Utilisation d'alias dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous CreateAliasà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Creates a custom alias for the specified target key asynchronously.
    *
    * @param targetKeyId the ID of the target key for the alias
    * @param aliasName   the name of the alias to create
    * @return a {@link CompletableFuture} that completes when the alias creation
operation is finished
    */
   public CompletableFuture<Void> createCustomAliasAsync(String targetKeyId,
String aliasName) {
       CreateAliasRequest aliasRequest = CreateAliasRequest.builder()
           .aliasName(aliasName)
           .targetKeyId(targetKeyId)
           .build();

       CompletableFuture<CreateAliasResponse> responseFuture =
getAsyncClient().createAlias(aliasRequest);
       responseFuture.whenComplete((response, exception) -> {
           if (exception == null) {
               logger.info("{} was successfully created.", aliasName);
           } else {
               if (exception instanceof ResourceExistsException) {
                   logger.info("Alias [{}] already exists. Moving on...",
aliasName);
               } else if (exception instanceof KmsException kmsEx) {
                   throw new RuntimeException("KMS error occurred while creating
alias: " + kmsEx.getMessage(), kmsEx);
               } else {
                   throw new RuntimeException("An unexpected error occurred
while creating alias: " + exception.getMessage(), exception);
               }
           }
```

```
        });

        return responseFuture.thenApply(response -> null);
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateAliasà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createCustomAlias(
    targetKeyIdVal: String?,
    aliasNameVal: String?,
) {
    val request =
        CreateAliasRequest {
            aliasName = aliasNameVal
            targetKeyId = targetKeyIdVal
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        kmsClient.createAlias(request)
        println("$aliasNameVal was successfully created")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateAliasà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param string $alias
     * @return void
     */
    public function createAlias(string $keyId, string $alias)
    {
        try{
            $this->client->createAlias([
                'TargetKeyId' => $keyId,
                'AliasName' => $alias,
            ]);
        }catch (KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidAliasNameException"){
                echo "The request was rejected because the specified alias name
is not valid.";
            }
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateAliasà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def create_alias(self, key_id: str, alias: str) -> None:
        """
        Creates an alias for the specified key.

        :param key_id: The ARN or ID of a key to give an alias.
        :param alias: The alias to assign to the key.
        """
        try:
            self.kms_client.create_alias(AliasName=alias, TargetKeyId=key_id)
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExistsException":
                logger.error(
                    "Could not create the alias %s because it already exists.",
 key_id
                )
```

```
            else:
                logger.error(
                    "Couldn't encrypt text. Here's why: %s",
                    err.response["Error"]["Message"],
                )
                raise
```

- Pour plus de détails sur l'API, consultez CreateAliasle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateGrant`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `CreateGrant`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        public static async Task Main()
        {
            var client = new AmazonKeyManagementServiceClient();
```

```
            // The identity that is given permission to perform the operations
            // specified in the grant.
            var grantee = "arn:aws:iam::111122223333:role/ExampleRole";

            // The identifier of the AWS KMS key to which the grant applies. You
            // can use the key ID or the Amazon Resource Name (ARN) of the KMS
key.
            var keyId = "7c9eccc2-38cb-4c4f-9db3-766ee8dd3ad4";

            var request = new CreateGrantRequest
            {
                GranteePrincipal = grantee,
                KeyId = keyId,

                // A list of operations that the grant allows.
                Operations = new List<string>
                {
                    "Encrypt",
                    "Decrypt",
                },
            };

            var response = await client.CreateGrantAsync(request);

            string grantId = response.GrantId; // The unique identifier of the
grant.
            string grantToken = response.GrantToken; // The grant token.

            Console.WriteLine($"Id: {grantId}, Token: {grantToken}");
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateGrantà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour créer une subvention

L'`create-grant`exemple suivant crée une autorisation qui permet à l'`exampleUser`utilisateur d'utiliser la `decrypt` commande sur l'`1234abcd-12ab-34cd-56ef-1234567890ab`exemple de clé KMS. Le principal partant à la retraite est le `adminRole` rôle. L'autorisation utilise la `EncryptionContextSubset` contrainte d'autorisation pour autoriser cette autorisation uniquement lorsque le contexte de chiffrement de la `decrypt` demande inclut la paire `"Department": "IT"` clé-valeur.

```
aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::123456789012:user/exampleUser \
    --operations Decrypt \
    --constraints EncryptionContextSubset={Department=IT} \
    --retiring-principal arn:aws:iam::123456789012:role/adminRole
```

Sortie :

```
{
    "GrantId":
 "1a2b3c4d2f5e69f440bae30eaec9570bb1fb7358824f9ddfa1aa5a0dab1a59b2",
    "GrantToken": "<grant token here>"
}
```

Pour afficher des informations détaillées sur la subvention, utilisez la `list-grants` commande.

Pour plus d'informations, consultez la section Subventions dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous CreateGrantà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Grants permissions to a specified principal on a customer master key (CMK)
asynchronously.
    *
    * @param keyId             The unique identifier for the customer master key
(CMK) that the grant applies to.
    * @param granteePrincipal  The principal that is given permission to perform
the operations that the grant permits on the CMK.
    * @return A {@link CompletableFuture} that, when completed, contains the ID
of the created grant.
    * @throws RuntimeException If an error occurs during the grant creation
process.
    */
   public CompletableFuture<String> grantKeyAsync(String keyId, String
granteePrincipal) {
       List<GrantOperation> grantPermissions = List.of(
           GrantOperation.ENCRYPT,
           GrantOperation.DECRYPT,
           GrantOperation.DESCRIBE_KEY
       );

       CreateGrantRequest grantRequest = CreateGrantRequest.builder()
           .keyId(keyId)
           .name("grant1")
           .granteePrincipal(granteePrincipal)
           .operations(grantPermissions)
           .build();

       CompletableFuture<CreateGrantResponse> responseFuture =
getAsyncClient().createGrant(grantRequest);
       responseFuture.whenComplete((response, ex) -> {
           if (ex == null) {
               logger.info("Grant created successfully with ID: " +
response.grantId());
           } else {
               if (ex instanceof KmsException kmsEx) {
                   throw new RuntimeException("Failed to create grant: " +
kmsEx.getMessage(), kmsEx);
               } else {
                   throw new RuntimeException("An unexpected error occurred: " +
ex.getMessage(), ex);
               }
           }
```

```
        });

        return responseFuture.thenApply(CreateGrantResponse::grantId);
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateGrantà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createNewGrant(
    keyIdVal: String?,
    granteePrincipalVal: String?,
    operation: String,
): String? {
    val operationOb = GrantOperation.fromValue(operation)
    val grantOperationList = ArrayList<GrantOperation>()
    grantOperationList.add(operationOb)

    val request =
        CreateGrantRequest {
            keyId = keyIdVal
            granteePrincipal = granteePrincipalVal
            operations = grantOperationList
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.createGrant(request)
        return response.grantId
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateGrantà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /***
    * @param string $keyId
    * @param string $granteePrincipal
    * @param array $operations
    * @param array $grantTokens
    * @return Result
    */
   public function createGrant(string $keyId, string $granteePrincipal, array
$operations, array $grantTokens = [])
   {
       $args = [
           'KeyId' => $keyId,
           'GranteePrincipal' => $granteePrincipal,
           'Operations' => $operations,
       ];
       if($grantTokens){
           $args['GrantTokens'] = $grantTokens;
       }
       try{
           return $this->client->createGrant($args);
       }catch(KmsException $caught){
           if($caught->getAwsErrorMessage() == "InvalidGrantTokenException"){
               echo "The request was rejected because the specified grant token
is not valid.\n";
           }
           throw $caught;
       }
```

```
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateGrantà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def create_grant(
        self, key_id: str, principal: str, operations: [str]
    ) -> dict[str, str]:
        """
        Creates a grant for a key that lets a principal generate a symmetric data
        encryption key.

        :param key_id: The ARN or ID of the key.
```

```
        :param principal: The principal to grant permission to.
        :param operations: The operations to grant permission for.
        :return: The grant that is created.
        """
        try:
            return self.kms_client.create_grant(
                KeyId=key_id,
                GranteePrincipal=principal,
                Operations=operations,
            )
        except ClientError as err:
            logger.error(
                "Couldn't create a grant on key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez CreateGrantle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `CreateKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Principes de base
- Travaillez avec le chiffrement des tables

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.KeyManagementService;
    using Amazon.KeyManagementService.Model;

    /// <summary>
    /// Shows how to create a new AWS Key Management Service (AWS KMS)
    /// key.
    /// </summary>
    public class CreateKey
    {
        public static async Task Main()
        {
            // Note that if you need to create a Key in an AWS Region
            // other than the Region defined for the default user, you need to
            // pass the Region to the client constructor.
            var client = new AmazonKeyManagementServiceClient();

            // The call to CreateKeyAsync will create a symmetrical AWS KMS
            // key. For more information about symmetrical and asymmetrical
            // keys, see:
            //
            // https://docs.aws.amazon.com/kms/latest/developerguide/symm-asymm-
choose.html
            var response = await client.CreateKeyAsync(new CreateKeyRequest());

            // The KeyMetadata object contains information about the new AWS KMS
 key.
            KeyMetadata keyMetadata = response.KeyMetadata;

            if (keyMetadata is not null)
            {
```

```
                Console.WriteLine($"KMS Key: {keyMetadata.KeyId} was successfully
created.");
            }
            else
            {
                Console.WriteLine("Could not create KMS Key.");
            }
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Exemple 1 : pour créer une clé KMS gérée par le client dans AWS KMS

L'`create-key`exemple suivant crée une clé KMS de chiffrement symétrique.

Pour créer la clé KMS de base, une clé de chiffrement symétrique, il n'est pas nécessaire de spécifier de paramètres. Les valeurs par défaut de ces paramètres créent une clé de chiffrement symétrique.

Comme cette commande ne spécifie pas de politique de clé, la clé KMS obtient la politique de clé par défaut pour les clés KMS créées par programmation. Pour afficher la politique clé, utilisez la `get-key-policy` commande. Pour modifier la politique clé, utilisez la `put-key-policy` commande.

```
aws kms create-key
```

La `create-key` commande renvoie les métadonnées clés, y compris l'ID de clé et l'ARN de la nouvelle clé KMS. Vous pouvez utiliser ces valeurs pour identifier la clé KMS dans d'autres opérations AWS KMS. La sortie n'inclut pas les balises. Pour afficher les balises d'une clé KMS, utilisez le`list-resource-tags command`.

Sortie :

```
{
    "KeyMetadata": {
        "AWSAccountId": "111122223333",
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": "2017-07-05T14:04:55-07:00",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "AWS_KMS"
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

Remarque : La `create-key` commande ne vous permet pas de spécifier un alias. Pour créer un alias pour la nouvelle clé KMS, utilisez la `create-alias` commande.

Pour plus d'informations, consultez la section Création de clés dans le Guide du développeur du service de gestion des AWS clés.

Exemple 2 : pour créer une clé RSA KMS asymétrique pour le chiffrement et le déchiffrement

L'`create-key`exemple suivant crée une clé KMS contenant une paire de clés RSA asymétrique pour le chiffrement et le déchiffrement. Les spécifications et l'utilisation de la clé ne peuvent pas être modifiées une fois la clé créée. :

```
aws kms create-key \
   --key-spec RSA_4096 \
   --key-usage ENCRYPT_DECRYPT
```

Sortie :

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2021-04-05T14:04:55-07:00",
        "CustomerMasterKeySpec": "RSA_4096",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
            "RSAES_OAEP_SHA_1",
            "RSAES_OAEP_SHA_256"
        ],
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "RSA_4096",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "AWS_KMS"
    }
}
```

Pour plus d'informations, consultez la section Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

Exemple 3 : pour créer une clé KMS à courbe elliptique asymétrique pour la signature et la vérification

Pour créer une clé KMS asymétrique contenant une paire de clés à courbe elliptique asymétrique (ECC) pour la signature et la vérification. Le `--key-usage` paramètre est obligatoire même s'il s'`SIGN_VERIFY`agit de la seule valeur valide pour les clés ECC KMS. Les spécifications et l'utilisation de la clé ne peuvent pas être modifiées une fois la clé créée. :

```
aws kms create-key \
    --key-spec ECC_NIST_P521 \
    --key-usage SIGN_VERIFY
```

Sortie :

```
{
```

```
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2019-12-02T07:48:55-07:00",
        "CustomerMasterKeySpec": "ECC_NIST_P521",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "ECC_NIST_P521",
        "KeyState": "Enabled",
        "KeyUsage": "SIGN_VERIFY",
        "MultiRegion": false,
        "Origin": "AWS_KMS",
        "SigningAlgorithms": [
            "ECDSA_SHA_512"
        ]
    }
}
```

Pour plus d'informations, consultez la section Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

Exemple 4 : pour créer une clé KMS ML-DSA asymétrique pour la signature et la vérification

Cet exemple crée une clé ML-DSA (Module-Lattice Digital Signature Algorithm) pour la signature et la vérification. Le paramètre d'utilisation des clés est obligatoire même s'il s'`SIGN_VERIFY`agit de la seule valeur valide pour les clés ML-DSA.

```
aws kms create-key \
    --key-spec ML_DSA_65 \
    --key-usage SIGN_VERIFY
```

Sortie :

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2019-12-02T07:48:55-07:00",
```

```
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "ML_DSA_65",
        "KeyState": "Enabled",
        "KeyUsage": "SIGN_VERIFY",
        "MultiRegion": false,
        "Origin": "AWS_KMS",
        "SigningAlgorithms": [
            "ML_DSA_SHAKE_256"
        ]
    }
}
```

Pour plus d'informations, consultez la section Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

Exemple 5 : pour créer une clé HMAC KMS

L'`create-key`exemple suivant crée une clé HMAC KMS 384 bits. La `GENERATE_VERIFY_MAC` valeur du `--key-usage` paramètre est obligatoire même s'il s'agit de la seule valeur valide pour les clés HMAC KMS.

```
aws kms create-key \
    --key-spec HMAC_384 \
    --key-usage GENERATE_VERIFY_MAC
```

Sortie :

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2022-04-05T14:04:55-07:00",
        "CustomerMasterKeySpec": "HMAC_384",
        "Description": "",
        "Enabled": true,
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "HMAC_384",
```

```
        "KeyState": "Enabled",
        "KeyUsage": "GENERATE_VERIFY_MAC",
        "MacAlgorithms": [
            "HMAC_SHA_384"
        ],
        "MultiRegion": false,
        "Origin": "AWS_KMS"
    }
}
```

Pour plus d'informations, consultez la section Clés HMAC dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

Exemple 6 : pour créer une clé KMS primaire multirégionale

L'`create-key`exemple suivant crée une clé de chiffrement symétrique principale multirégionale. Étant donné que les valeurs par défaut de tous les paramètres créent une clé de chiffrement symétrique, seul le `--multi-region` paramètre est requis pour cette clé KMS. Dans la AWS CLI, pour indiquer qu'un paramètre booléen est vrai, il suffit de spécifier le nom du paramètre.

```
aws kms create-key \
    --multi-region
```

Sortie :

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef12345678990ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2021-09-02T016:15:21-09:00",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
          "SYMMETRIC_DEFAULT"
        ],
        "KeyId": "mrk-1234abcd12ab34cd56ef12345678990ab",
```

```
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": true,
        "MultiRegionConfiguration": {
            "MultiRegionKeyType": "PRIMARY",
            "PrimaryKey": {
                "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef12345678990ab",
                "Region": "us-west-2"
            },
            "ReplicaKeys": []
        },
        "Origin": "AWS_KMS"
    }
}
```

Pour plus d'informations, consultez la section Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

Exemple 7 : Pour créer une clé KMS pour le matériel clé importé

L'`create-key`exemple suivant crée une clé KMS sans aucun élément clé. Lorsque l'opération est terminée, vous pouvez importer votre propre matériel clé dans la clé KMS. Pour créer cette clé KMS, définissez le `--origin` paramètre sur`EXTERNAL`.

```
aws kms create-key \
    --origin EXTERNAL
```

Sortie :

```
{
     "KeyMetadata": {
         "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
         "AWSAccountId": "111122223333",
         "CreationDate": "2019-12-02T07:48:55-07:00",
         "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
         "Description": "",
         "Enabled": false,
         "EncryptionAlgorithms": [
```

```
            "SYMMETRIC_DEFAULT"
        ],
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "PendingImport",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "EXTERNAL"
    }
}
```

Pour plus d'informations, consultez la section Importation de matériel clé dans les clés AWS KMS dans le guide du développeur du service de gestion des AWS clés.

Exemple 6 : pour créer une clé KMS dans un magasin de clés AWS CloudHSM

L'`create-key`exemple suivant crée une clé KMS dans le magasin de clés AWS CloudHSM spécifié. L'opération crée la clé KMS et ses métadonnées dans AWS KMS et crée le matériel clé dans le cluster AWS CloudHSM associé au magasin de clés personnalisé. Les paramètres `--custom-key-store-id` et `--origin` sont obligatoires.

```
aws kms create-key \
    --origin AWS_CLOUDHSM \
    --custom-key-store-id cks-1234567890abcdef0
```

Sortie :

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CloudHsmClusterId": "cluster-1a23b4cdefg",
        "CreationDate": "2019-12-02T07:48:55-07:00",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "CustomKeyStoreId": "cks-1234567890abcdef0",
        "Description": "",
        "Enabled": true,
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
```

```
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "AWS_CLOUDHSM"
    }
}
```

Pour plus d'informations, consultez les magasins de clés AWS CloudHSM dans AWS le Guide du développeur du service de gestion des clés.

Exemple 8 : pour créer une clé KMS dans un magasin de clés externe

L'`create-key`exemple suivant crée une clé KMS dans le magasin de clés externe spécifié. Les `--xks-key-id` paramètres `--custom-key-store-id--origin`, et sont obligatoires dans cette commande.

Le `--xks-key-id` paramètre indique l'ID d'une clé de chiffrement symétrique existante dans votre gestionnaire de clés externe. Cette clé sert de matériau de clé externe pour la clé KMS. La valeur du `--origin` paramètre doit être `EXTERNAL_KEY_STORE` .Le `custom-key-store-id` paramètre doit identifier un magasin de clés externe connecté à son proxy de magasin de clés externe.

```
aws kms create-key \
    --origin EXTERNAL_KEY_STORE \
    --custom-key-store-id cks-9876543210fedcba9 \
    --xks-key-id bb8562717f809024
```

Sortie :

```
{
    "KeyMetadata": {
        "Arn": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "AWSAccountId": "111122223333",
        "CreationDate": "2022-12-02T07:48:55-07:00",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "CustomKeyStoreId": "cks-9876543210fedcba9",
        "Description": "",
```

```
        "Enabled": true,
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeySpec": "SYMMETRIC_DEFAULT",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "MultiRegion": false,
        "Origin": "EXTERNAL_KEY_STORE",
        "XksKeyConfiguration": {
            "Id": "bb8562717f809024"
        }
    }
}
```

Pour plus d'informations, consultez la section Stockages de clés externes dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous CreateKeyà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Creates a new symmetric encryption key asynchronously.
     *
     * @param keyDesc the description of the key to be created
     * @return a {@link CompletableFuture} that completes with the ID of the
newly created key
     * @throws RuntimeException if an error occurs while creating the key
     */
    public CompletableFuture<String> createKeyAsync(String keyDesc) {
```

```
        CreateKeyRequest keyRequest = CreateKeyRequest.builder()
            .description(keyDesc)
            .keySpec(KeySpec.SYMMETRIC_DEFAULT)
            .keyUsage(KeyUsageType.ENCRYPT_DECRYPT)
            .build();

        return getAsyncClient().createKey(keyRequest)
            .thenApply(resp -> resp.keyMetadata().keyId())
            .exceptionally(ex -> {
                throw new RuntimeException("An error occurred while creating the
 key: " + ex.getMessage(), ex);
            });
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createKey(keyDesc: String?): String? {
    val request =
        CreateKeyRequest {
            description = keyDesc
            customerMasterKeySpec = CustomerMasterKeySpec.SymmetricDefault
            keyUsage = KeyUsageType.fromValue("ENCRYPT_DECRYPT")
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val result = kmsClient.createKey(request)
        println("Created a customer key with id " + result.keyMetadata?.arn)
        return result.keyMetadata?.keyId
    }
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /***
    * @param string $keySpec
    * @param string $keyUsage
    * @param string $description
    * @return array
    */
   public function createKey(string $keySpec = "", string $keyUsage = "", string
$description = "Created by the SDK for PHP")
   {
       $parameters = ['Description' => $description];
       if($keySpec && $keyUsage){
           $parameters['KeySpec'] = $keySpec;
           $parameters['KeyUsage'] = $keyUsage;
       }
       try {
           $result = $this->client->createKey($parameters);
           return $result['KeyMetadata'];
       }catch(KmsException $caught){
           // Check for error specific to createKey operations
           if ($caught->getAwsErrorMessage() == "LimitExceededException"){
               echo "The request was rejected because a quota was exceeded. For
more information, see Quotas in the Key Management Service Developer Guide.";
           }
           throw $caught;
```

```
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def create_key(self, key_description: str) -> dict[str, any]:
        """
        Creates a key with a user-provided description.

        :param key_description: A description for the key.
        :return: The key ID.
```

```
        """
        try:
            key = self.kms_client.create_key(Description=key_description)
["KeyMetadata"]
            self.created_keys.append(key)
            return key
        except ClientError as err:
            logging.error(
                "Couldn't create your key. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez CreateKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'

# Create a AWS KMS key.
# As long we are only encrypting small amounts of data (4 KiB or less) directly,
# a KMS key is fine for our purposes.
# For larger amounts of data,
# use the KMS key to encrypt a data encryption key (DEK).

client = Aws::KMS::Client.new

resp = client.create_key({
                           tags: [
```

```
                                {
                                  tag_key: 'CreatedBy',
                                  tag_value: 'ExampleUser'
                                }
                              ]
                            })

puts resp.key_metadata.key_id
```

- Pour plus de détails sur l'API, reportez-vous CreateKeyà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn make_key(client: &Client) -> Result<(), Error> {
    let resp = client.create_key().send().await?;

    let id = resp.key_metadata.as_ref().unwrap().key_id();

    println!("Key: {}", id);

    Ok(())
}
```

- Pour plus de détails sur l'API, voir CreateKeyla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`Decrypt`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `Decrypt`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Exemple 1 : pour déchiffrer un message chiffré avec une clé KMS symétrique (Linux et macOS)

L'exemple de `decrypt` commande suivant illustre la méthode recommandée pour déchiffrer des données à l'aide de la AWS CLI. Cette version montre comment déchiffrer des données sous une clé KMS symétrique.

Fournissez le texte chiffré dans un fichier. Dans la valeur du `--ciphertext-blob` paramètre, utilisez le `fileb://` préfixe, qui indique à la CLI de lire les données d'un fichier binaire. Si le fichier ne se trouve pas dans le répertoire actuel, saisissez le chemin complet du fichier. Pour plus d'informations sur la lecture des valeurs des paramètres de la AWS CLI depuis un fichier, consultez la section Chargement des paramètres de la AWS CLI depuis un fichier < https://docs.aws.amazon.com/cli/ latest/userguide/cli - usage-parameters-file .html> dans le guide de l'utilisateur de l'interface de ligne de AWS commande et les meilleures pratiques pour les paramètres des fichiers locaux < https://aws.amazon.com/blogs/ developer/ best-practices-for-local -file-parameters/> dans le blog des outils de ligne de AWS commande. Spécifiez la clé KMS pour déchiffrer le texte chiffré. Le paramètre n'est pas obligatoire lors du déchiffrement avec une clé KMS symétrique. `--key-id` AWS KMS peut obtenir l'ID de la clé KMS qui a été utilisée pour chiffrer les données à partir des métadonnées du texte chiffré. Toutefois, la spécification de la clé KMS que vous utilisez est une bonne pratique. Cette pratique garantit que vous utilisez la clé KMS comme vous le souhaitez et vous empêche de déchiffrer par inadvertance un texte chiffré à l'aide d'une clé KMS non fiable.Demandez la sortie en texte brut sous forme de valeur de texte.Le paramètre `--query` indique à la CLI de n'obtenir que la valeur du champ à partir de la sortie. `Plaintext` Le `--output` paramètre renvoie la sortie sous forme de texte.Base64 décodez le texte en clair et enregistrez-le

dans un fichier. L'exemple suivant montre comment rediriger (|) la valeur du `Plaintext` paramètre vers l'utilitaire Base64, qui le décode. Ensuite, il redirige (>) la sortie décodée vers le `ExamplePlaintext` fichier.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte.

```
aws kms decrypt \
    --ciphertext-blob fileb://ExampleEncryptedFile \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --output text \
    --query Plaintext | base64 \
    --decode > ExamplePlaintextFile
```

Cette commande ne produit aucun résultat. La sortie de la `decrypt` commande est décodée en base64 et enregistrée dans un fichier.

Pour plus d'informations, voir Déchiffrer dans le manuel de référence de l'API du service de gestion des AWS clés.

Exemple 2 : pour déchiffrer un message chiffré avec une clé KMS symétrique (invite de commande Windows)

L'exemple suivant est identique au précédent, sauf qu'il utilise l'`certutil`utilitaire pour décoder les données en texte brut en Base64. Cette procédure nécessite deux commandes, comme indiqué dans les exemples suivants.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte.

```
aws kms decrypt ^
    --ciphertext-blob fileb://ExampleEncryptedFile ^
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^
    --output text ^
    --query Plaintext > ExamplePlaintextFile.base64
```

Exécutez la commande `certutil`.

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

Sortie :

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

Pour plus d'informations, voir Déchiffrer dans le manuel de référence de l'API du service de gestion des AWS clés.

Exemple 3 : pour déchiffrer un message chiffré avec une clé KMS asymétrique (Linux et macOS)

L'exemple de `decrypt` commande suivant montre comment déchiffrer des données chiffrées sous une clé KMS asymétrique RSA.

Lors de l'utilisation d'une clé KMS asymétrique, le `encryption-algorithm` paramètre, qui spécifie l'algorithme utilisé pour chiffrer le texte en clair, est obligatoire.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte.

```
aws kms decrypt \
    --ciphertext-blob fileb://ExampleEncryptedFile \
    --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --encryption-algorithm RSAES_OAEP_SHA_256 \
    --output text \
    --query Plaintext | base64 \
    --decode > ExamplePlaintextFile
```

Cette commande ne produit aucun résultat. La sortie de la `decrypt` commande est décodée en base64 et enregistrée dans un fichier.

Pour plus d'informations, consultez la section Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, voir Déchiffrer dans le manuel de référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Asynchronously decrypts the given encrypted data using the specified key
ID.
    *
    * @param encryptedData The encrypted data to be decrypted.
    * @param keyId The ID of the key to be used for decryption.
    * @return A CompletableFuture that, when completed, will contain the
decrypted data as a String.
    *         If an error occurs during the decryption process, the
CompletableFuture will complete
    *         exceptionally with the error, and the method will return an empty
String.
    */
   public CompletableFuture<String> decryptDataAsync(SdkBytes encryptedData,
String keyId) {
       DecryptRequest decryptRequest = DecryptRequest.builder()
           .ciphertextBlob(encryptedData)
           .keyId(keyId)
           .build();

       CompletableFuture<DecryptResponse> responseFuture =
getAsyncClient().decrypt(decryptRequest);
       responseFuture.whenComplete((decryptResponse, exception) -> {
           if (exception == null) {
               logger.info("Data decrypted successfully for key ID: " + keyId);
           } else {
               if (exception instanceof KmsException kmsEx) {
                   throw new RuntimeException("KMS error occurred while
decrypting data: " + kmsEx.getMessage(), kmsEx);
               } else {
                   throw new RuntimeException("An unexpected error occurred
while decrypting data: " + exception.getMessage(), exception);
```

```
                }
            }
        });

        return responseFuture.thenApply(decryptResponse ->
 decryptResponse.plaintext().asString(StandardCharsets.UTF_8));
    }
```

- Pour plus de détails sur l'API, voir Déchiffrer dans le guide de référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun encryptData(keyIdValue: String): ByteArray? {
    val text = "This is the text to encrypt by using the AWS KMS Service"
    val myBytes: ByteArray = text.toByteArray()

    val encryptRequest =
        EncryptRequest {
            keyId = keyIdValue
            plaintext = myBytes
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.encrypt(encryptRequest)
        val algorithm: String = response.encryptionAlgorithm.toString()
        println("The encryption algorithm is $algorithm")

        // Return the encrypted data.
        return response.ciphertextBlob
    }
}
```

```
suspend fun decryptData(
    encryptedDataVal: ByteArray?,
    keyIdVal: String?,
) {
    val decryptRequest =
        DecryptRequest {
            ciphertextBlob = encryptedDataVal
            keyId = keyIdVal
        }
    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val decryptResponse = kmsClient.decrypt(decryptRequest)
        val myVal = decryptResponse.plaintext

        // Print the decrypted data.
        print(myVal)
    }
}
```

- Pour plus de détails sur l'API, voir Décrypter dans le AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param string $ciphertext
     * @param string $algorithm
     * @return Result
     */
```

```
    public function decrypt(string $keyId, string $ciphertext, string $algorithm
 = "SYMMETRIC_DEFAULT")
    {
        try{
            return $this->client->decrypt([
                'CiphertextBlob' => $ciphertext,
                'EncryptionAlgorithm' => $algorithm,
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem decrypting the data: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, voir Déchiffrer dans le guide de référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.
```

```
        :return: An instance of KeyEncrypt initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def decrypt(self, key_id: str, cipher_text: str) -> bytes:
        """
        Decrypts text previously encrypted with a key.

        :param key_id: The ARN or ID of the key used to decrypt the data.
        :param cipher_text: The encrypted text to decrypt.
        :return: The decrypted text.
        """
        try:
            return self.kms_client.decrypt(KeyId=key_id,
CiphertextBlob=cipher_text)[
                "Plaintext"
            ]
        except ClientError as err:
            logger.error(
                "Couldn't decrypt your ciphertext. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API Decrypt in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'

# Decrypted blob

blob =
 '01020200785d68faeec386af1057904926253051eb2919d3c16078badf65b808b26dd057c101747cadf3593
blob_packed = [blob].pack('H*')

client = Aws::KMS::Client.new(region: 'us-west-2')

resp = client.decrypt({
                        ciphertext_blob: blob_packed
                      })

puts 'Raw text: '
puts resp.plaintext
```

- Pour plus de détails sur l'API, voir Déchiffrer dans le guide de référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn decrypt_key(client: &Client, key: &str, filename: &str) -> Result<(),
 Error> {
    // Open input text file and get contents as a string
    // input is a base-64 encoded string, so decode it:
    let data = fs::read_to_string(filename)
        .map(|input| {
            base64::decode(input).expect("Input file does not contain valid base
 64 characters.")
        })
```

```
        .map(Blob::new);

    let resp = client
        .decrypt()
        .key_id(key)
        .ciphertext_blob(data.unwrap())
        .send()
        .await?;

    let inner = resp.plaintext.unwrap();
    let bytes = inner.as_ref();

    let s = String::from_utf8(bytes.to_vec()).expect("Could not convert to
 UTF-8");

    println!();
    println!("Decoded string:");
    println!("{}", s);

    Ok(())
}
```

- Pour plus de détails sur l'API, voir Decrypt in AWS SDK for Rust API reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteAlias`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteAlias`.

CLI

AWS CLI

Pour supprimer un alias AWS KMS

L'`delete-alias`exemple suivant supprime l'alias`alias/example-alias`. Le nom de l'alias doit commencer par alias/.

```
aws kms delete-alias \
```

```
    --alias-name alias/example-alias
```

Cette commande ne produit aucun résultat. Pour trouver l'alias, utilisez la `list-aliases` commande.

Pour plus d'informations, consultez la section Suppression d'un alias dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous DeleteAliasà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Deletes a specific KMS alias asynchronously.
    *
    * @param aliasName the name of the alias to be deleted
    * @return a {@link CompletableFuture} representing the asynchronous
operation of deleting the specified alias
    */
   public CompletableFuture<Void> deleteSpecificAliasAsync(String aliasName) {
       DeleteAliasRequest deleteAliasRequest = DeleteAliasRequest.builder()
           .aliasName(aliasName)
           .build();

       return getAsyncClient().deleteAlias(deleteAliasRequest)
           .thenRun(() -> {
               logger.info("Alias {} has been deleted successfully", aliasName);
           })
           .exceptionally(throwable -> {
               throw new RuntimeException("Failed to delete alias: " +
aliasName, throwable);
           });
```

```
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteAliasà la section Référence des AWS SDK for Java 2.x API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $aliasName
     * @return void
     */
    public function deleteAlias(string $aliasName)
    {
        try {
            $this->client->deleteAlias([
                'AliasName' => $aliasName,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem deleting the alias: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteAliasà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def delete_alias(self, alias: str) -> None:
        """
        Deletes an alias.

        :param alias: The alias to delete.
        """
        try:
            self.kms_client.delete_alias(AliasName=alias)
        except ClientError as err:
            logger.error(
                "Couldn't delete alias %s. Here's why: %s",
                alias,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteAliasle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DescribeKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DescribeKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Retrieve information about an AWS Key Management Service (AWS KMS) key.
/// You can supply either the key Id or the key Amazon Resource Name (ARN)
/// to the DescribeKeyRequest KeyId property.
```

```
    /// </summary>
    public class DescribeKey
    {
        public static async Task Main()
        {
            var keyId = "7c9eccc2-38cb-4c4f-9db3-766ee8dd3ad4";
            var request = new DescribeKeyRequest
            {
                KeyId = keyId,
            };

            var client = new AmazonKeyManagementServiceClient();

            var response = await client.DescribeKeyAsync(request);
            var metadata = response.KeyMetadata;

            Console.WriteLine($"{metadata.KeyId} created on:
{metadata.CreationDate}");
            Console.WriteLine($"State: {metadata.KeyState}");
            Console.WriteLine($"{metadata.Description}");
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DescribeKeyà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Exemple 1 : pour trouver des informations détaillées sur une clé KMS

L'`describe-key`exemple suivant fournit des informations détaillées sur la clé AWS gérée pour Amazon S3 dans l'exemple de compte et de région. Vous pouvez utiliser cette commande pour obtenir des informations sur les clés AWS gérées et les clés gérées par le client.

Pour spécifier la clé KMS, utilisez le `key-id` paramètre. Cet exemple utilise une valeur de nom d'alias, mais vous pouvez utiliser un ID de clé, un ARN de clé, un nom d'alias ou un ARN d'alias dans cette commande.

```
aws kms describe-key \
    --key-id alias/aws/s3
```

Sortie :

```
{
    "KeyMetadata": {
        "AWSAccountId": "846764612917",
        "KeyId": "b8a9477d-836c-491f-857e-07937918959b",
        "Arn": "arn:aws:kms:us-west-2:846764612917:key/
b8a9477d-836c-491f-857e-07937918959b",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "CreationDate": 2017-06-30T21:44:32.140000+00:00,
        "Enabled": true,
        "Description": "Default KMS key that protects my S3 objects when no other
 key is defined",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "AWS",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ]
    }
}
```

Pour plus d'informations, consultez la section Affichage des clés dans le Guide du développeur du service de gestion des AWS clés.

Exemple 2 : pour obtenir des informations sur une clé KMS asymétrique RSA

L'`describe-key`exemple suivant fournit des informations détaillées sur une clé RSA KMS asymétrique utilisée pour la signature et la vérification.

```
aws kms describe-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Sortie :

```
{
```

```
    "KeyMetadata": {
        "AWSAccountId": "111122223333",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": "2019-12-02T19:47:14.861000+00:00",
        "CustomerMasterKeySpec": "RSA_2048",
        "Enabled": false,
        "Description": "",
        "KeyState": "Disabled",
        "Origin": "AWS_KMS",
        "MultiRegion": false,
        "KeyManager": "CUSTOMER",
        "KeySpec": "RSA_2048",
        "KeyUsage": "SIGN_VERIFY",
        "SigningAlgorithms": [
            "RSASSA_PKCS1_V1_5_SHA_256",
            "RSASSA_PKCS1_V1_5_SHA_384",
            "RSASSA_PKCS1_V1_5_SHA_512",
            "RSASSA_PSS_SHA_256",
            "RSASSA_PSS_SHA_384",
            "RSASSA_PSS_SHA_512"
        ]
    }
}
```

Exemple 3 : pour obtenir des informations sur une clé de réplique multirégionale

L'`describe-key`exemple suivant permet d'obtenir les métadonnées d'une clé de réplique multirégionale. Cette clé multirégionale est une clé de chiffrement symétrique. La sortie d'une `describe-key` commande pour une clé multirégionale renvoie des informations sur la clé primaire et toutes ses répliques.

```
aws kms describe-key \
    --key-id arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab
```

Sortie :

```
{
    "KeyMetadata": {
        "MultiRegion": true,
```

```
        "AWSAccountId": "111122223333",
        "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "CreationDate": "2021-06-28T21:09:16.114000+00:00",
        "CurrentKeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
        "Description": "",
        "Enabled": true,
        "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "KeyManager": "CUSTOMER",
        "KeyState": "Enabled",
        "KeyUsage": "ENCRYPT_DECRYPT",
        "Origin": "AWS_KMS",
        "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "EncryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "MultiRegionConfiguration": {
            "MultiRegionKeyType": "PRIMARY",
            "PrimaryKey": {
                "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                "Region": "us-west-2"
            },
            "ReplicaKeys": [
                {
                    "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "eu-west-1"
                },
                {
                    "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "ap-northeast-1"
                },
                {
                    "Arn": "arn:aws:kms:sa-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
                    "Region": "sa-east-1"
                }
            ]
        }
    }
```

```
}
```

Exemple 4 : pour obtenir des informations sur une clé HMAC KMS

L'`describe-key`exemple suivant permet d'obtenir des informations détaillées sur une clé HMAC KMS.

```
aws kms describe-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Sortie :

```
{
    "KeyMetadata": {
        "AWSAccountId": "123456789012",
        "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "Arn": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "CreationDate": "2022-04-03T22:23:10.194000+00:00",
        "Enabled": true,
        "Description": "Test key",
        "KeyUsage": "GENERATE_VERIFY_MAC",
        "KeyState": "Enabled",
        "Origin": "AWS_KMS",
        "KeyManager": "CUSTOMER",
        "CustomerMasterKeySpec": "HMAC_256",
        "MacAlgorithms": [
            "HMAC_SHA_256"
        ],
        "MultiRegion": false
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DescribeKeyà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Asynchronously checks if a specified key is enabled.
     *
     * @param keyId the ID of the key to check
     * @return a {@link CompletableFuture} that, when completed, indicates
whether the key is enabled or not
     *
     * @throws RuntimeException if an exception occurs while checking the key
state
     */
    public CompletableFuture<Boolean> isKeyEnabledAsync(String keyId) {
        DescribeKeyRequest keyRequest = DescribeKeyRequest.builder()
            .keyId(keyId)
            .build();

        CompletableFuture<DescribeKeyResponse> responseFuture =
getAsyncClient().describeKey(keyRequest);
        return responseFuture.whenComplete((resp, ex) -> {
            if (resp != null) {
                KeyState keyState = resp.keyMetadata().keyState();
                if (keyState == KeyState.ENABLED) {
                    logger.info("The key is enabled.");
                } else {
                    logger.info("The key is not enabled. Key state: {}",
keyState);
                }
            } else {
                throw new RuntimeException(ex);
            }
        }).thenApply(resp -> resp.keyMetadata().keyState() == KeyState.ENABLED);
    }
```

- Pour plus de détails sur l'API, reportez-vous DescribeKeyà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun describeSpecifcKey(keyIdVal: String?) {
    val request =
        DescribeKeyRequest {
            keyId = keyIdVal
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.describeKey(request)
        println("The key description is ${response.keyMetadata?.description}")
        println("The key ARN is ${response.keyMetadata?.arn}")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DescribeKeyà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @return array
     */
    public function describeKey(string $keyId)
    {
        try {
            $result = $this->client->describeKey([
                "KeyId" => $keyId,
            ]);
            return $result['KeyMetadata'];
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "The request was rejected because the specified entity or
 resource could not be found.\n";
            }
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DescribeKeyà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []
```

```
    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def describe_key(self, key_id: str) -> dict[str, any]:
        """
        Describes a key.

        :param key_id: The ARN or ID of the key to describe.
        :return: Information about the key.
        """

        try:
            key = self.kms_client.describe_key(KeyId=key_id)["KeyMetadata"]
            return key
        except ClientError as err:
            logging.error(
                "Couldn't get key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DescribeKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DisableKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DisableKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Disable an AWS Key Management Service (AWS KMS) key and then retrieve
/// the key's status to show that it has been disabled.
/// </summary>
public class DisableKey
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();

        // The identifier of the AWS KMS key to disable. You can use the
        // key Id or the Amazon Resource Name (ARN) of the AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";

        var request = new DisableKeyRequest
        {
```

```
                KeyId = keyId,
            };

            var response = await client.DisableKeyAsync(request);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                // Retrieve information about the key to show that it has now
                // been disabled.
                var describeResponse = await client.DescribeKeyAsync(new
DescribeKeyRequest
                {
                    KeyId = keyId,
                });
                Console.WriteLine($"{describeResponse.KeyMetadata.KeyId} - state:
{describeResponse.KeyMetadata.KeyState}");
            }
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DisableKeyà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour désactiver temporairement une clé KMS

La `disable-key` commande suivante désactive une clé KMS gérée par le client. Pour réactiver la clé KMS, utilisez la `enable-key` commande.

```
aws kms disable-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Activation et désactivation des clés dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous DisableKeyà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Asynchronously disables the specified AWS Key Management Service (KMS)
key.
    *
    * @param keyId the ID or Amazon Resource Name (ARN) of the KMS key to be
disabled
    * @return a CompletableFuture that, when completed, indicates that the key
has been disabled successfully
    */
   public CompletableFuture<Void> disableKeyAsync(String keyId) {
       DisableKeyRequest keyRequest = DisableKeyRequest.builder()
           .keyId(keyId)
           .build();

       return getAsyncClient().disableKey(keyRequest)
           .thenRun(() -> {
               logger.info("Key {} has been disabled successfully",keyId);
           })
           .exceptionally(throwable -> {
               throw new RuntimeException("Failed to disable key: " + keyId,
throwable);
           });
   }
```

- Pour plus de détails sur l'API, reportez-vous DisableKeyà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun disableKey(keyIdVal: String?) {
    val request =
        DisableKeyRequest {
            keyId = keyIdVal
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        kmsClient.disableKey(request)
        println("$keyIdVal was successfully disabled")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DisableKeyà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @return void
```

```
     */
    public function disableKey(string $keyId)
    {
        try {
            $this->client->disableKey([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem disabling the key: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DisableKeyà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
```

```
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def disable_key(self, key_id: str) -> None:
        try:
            self.kms_client.disable_key(KeyId=key_id)
        except ClientError as err:
            logging.error(
                "Couldn't disable key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DisableKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`EnableKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `EnableKey`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Enable an AWS Key Management Service (AWS KMS) key.
/// </summary>
public class EnableKey
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();

        // The identifier of the AWS KMS key to enable. You can use the
        // key Id or the Amazon Resource Name (ARN) of the AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";

        var request = new EnableKeyRequest
        {
            KeyId = keyId,
        };

        var response = await client.EnableKeyAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            // Retrieve information about the key to show that it has now
            // been enabled.
            var describeResponse = await client.DescribeKeyAsync(new
DescribeKeyRequest
            {
                KeyId = keyId,
```

```
                });
                Console.WriteLine($"{describeResponse.KeyMetadata.KeyId} - state:
{describeResponse.KeyMetadata.KeyState}");
            }
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous EnableKeyà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour activer une clé KMS

L'`enable-key`exemple suivant active une clé gérée par le client. Vous pouvez utiliser une commande comme celle-ci pour activer une clé KMS que vous avez temporairement désactivée à l'aide de la `disable-key` commande. Vous pouvez également l'utiliser pour activer une clé KMS qui est désactivée car sa suppression a été planifiée et la suppression a été annulée.

Pour spécifier la clé KMS, utilisez le `key-id` paramètre. Cet exemple utilise une valeur d'ID de clé, mais vous pouvez utiliser une valeur d'ID de clé ou une valeur d'ARN de clé dans cette commande.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un identifiant valide.

```
aws kms enable-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette commande ne produit aucun résultat. Pour vérifier que la clé KMS est activée, utilisez la `describe-key` commande. Consultez les valeurs des `Enabled` champs `KeyState` et dans la `describe-key` sortie.

Pour plus d'informations, consultez la section Activation et désactivation des clés dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous EnableKeyà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Asynchronously enables the specified key.
    *
    * @param keyId the ID of the key to enable
    * @return a {@link CompletableFuture} that completes when the key has been
enabled
    */
   public CompletableFuture<Void> enableKeyAsync(String keyId) {
       EnableKeyRequest enableKeyRequest = EnableKeyRequest.builder()
           .keyId(keyId)
           .build();

       CompletableFuture<EnableKeyResponse> responseFuture =
getAsyncClient().enableKey(enableKeyRequest);
       responseFuture.whenComplete((response, exception) -> {
           if (exception == null) {
               logger.info("Key with ID [{}] has been enabled.", keyId);
           } else {
               if (exception instanceof KmsException kmsEx) {
                   throw new RuntimeException("KMS error occurred while enabling
key: " + kmsEx.getMessage(), kmsEx);
               } else {
                   throw new RuntimeException("An unexpected error occurred
while enabling key: " + exception.getMessage(), exception);
               }
           }
       });
```

```
        return responseFuture.thenApply(response -> null);
    }
```

- Pour plus de détails sur l'API, reportez-vous EnableKeyà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun enableKey(keyIdVal: String?) {
    val request =
        EnableKeyRequest {
            keyId = keyIdVal
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        kmsClient.enableKey(request)
        println("$keyIdVal was successfully enabled.")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous EnableKeyà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @return void
     */
    public function enableKey(string $keyId)
    {
        try {
            $this->client->enableKey([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "The request was rejected because the specified entity or
resource could not be found.\n";
            }
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous EnableKeyà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def enable_key(self, key_id: str) -> None:
        """
        Enables a key. Gets the key state after each state change.

        :param key_id: The ARN or ID of the key to enable.
        """
        try:
            self.kms_client.enable_key(KeyId=key_id)
        except ClientError as err:
            logging.error(
                "Couldn't enable key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez EnableKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **EnableKeyRotation** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `EnableKeyRotation`.

CLI

AWS CLI

Pour activer la rotation automatique d'une clé KMS

L'`enable-key-rotation`exemple suivant permet la rotation automatique d'une clé KMS gérée par le client avec une période de rotation de 180 jours. La clé KMS fera l'objet d'une rotation d'un an (environ 365 jours) à compter de la date d'exécution de cette commande et chaque année par la suite.

Le `--key-id` paramètre identifie la clé KMS. Cet exemple utilise une valeur ARN clé, mais vous pouvez utiliser l'ID de clé ou l'ARN de la clé KMS. Le `--rotation-period-in-days` paramètre indique le nombre de jours entre chaque date de rotation. Spécifiez une valeur comprise entre 90 et 2560 jours. Si aucune valeur n'est spécifiée, la valeur par défaut est de 365 jours.

```
aws kms enable-key-rotation \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --rotation-period-in-days 180
```

Cette commande ne produit aucun résultat. Pour vérifier que la clé KMS est activée, utilisez la `get-key-rotation-status` commande.

Pour plus d'informations, voir Rotation des clés dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous EnableKeyRotationà la section Référence des AWS CLI commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def enable_key_rotation(self, key_id: str) -> None:
        """
        Enables rotation for a key.

        :param key_id: The ARN or ID of the key to enable rotation for.
        """
        try:
            self.kms_client.enable_key_rotation(KeyId=key_id)
        except ClientError as err:
            logging.error(
                "Couldn't enable rotation for key '%s'. Here's why: %s",
```

```
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez EnableKeyRotationle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`Encrypt`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `Encrypt`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Exemple 1 : chiffrer le contenu d'un fichier sous Linux ou macOS

La `encrypt` commande suivante montre la méthode recommandée pour chiffrer les données avec la AWS CLI.

```
aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob | base64 \
    --decode > ExampleEncryptedFile
```

La commande effectue plusieurs opérations :

Utilise le `--plaintext` paramètre pour indiquer les données à chiffrer. La valeur de ce paramètre doit être codée en base64. La valeur du `plaintext` paramètre doit être codée en base64, ou vous devez utiliser le préfixe`fileb://`, qui indique à la AWS CLI de lire les données binaires du fichier. Si le fichier ne se trouve pas dans le répertoire actuel, tapez le chemin complet du fichier. Par exemple, `fileb:///var/tmp/ExamplePlaintextFile` ou `fileb://C:\Temp\ExamplePlaintextFile`. Pour plus d'informations sur la lecture des valeurs des paramètres de la AWS CLI depuis un fichier, consultez la section AWS Chargement de paramètres depuis un fichierdans le guide de l'utilisateur de l'interface de ligne de AWS commande et les meilleures pratiques pour les paramètres de fichiers locaux sur le `--output` blog des outils de ligne de commande. Ces `--query` paramètres extraient les données chiffrées, appelées texte chiffré, de la sortie de la commande.Pour plus d'informations sur le contrôle de la sortie, voir Contrôle de la commande Sortie dans le guide de l'utilisateur de l'interface de ligne de AWS commande. Utilise l'`base64`utilitaire pour décoder la sortie extraite en données binaires. Le texte chiffré renvoyé par une commande réussie `encrypt` est du texte codé en base64. Vous devez décoder ce texte avant de pouvoir utiliser la AWS CLI pour le déchiffrer. Enregistre le texte chiffré binaire dans un fichier.La dernière partie de la commande (`> ExampleEncryptedFile`) enregistre le texte chiffré binaire dans un fichier pour faciliter le déchiffrement. Pour un exemple de commande utilisant la AWS CLI pour déchiffrer des données, consultez les exemples de déchiffrement.

Exemple 2 : utilisation de la AWS CLI pour chiffrer des données sous Windows

Cet exemple est identique au précédent, sauf qu'il utilise l'`certutil`outil au lieu de`base64`. Cette procédure nécessite deux commandes, comme indiqué dans l'exemple suivant.

```
aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob > C:\Temp\ExampleEncryptedFile.base64

certutil -decode C:\Temp\ExampleEncryptedFile.base64 C:\Temp\ExampleEncryptedFile
```

Exemple 3 : Chiffrement avec une clé KMS asymétrique

La `encrypt` commande suivante montre comment chiffrer du texte en clair avec une clé KMS asymétrique. Le paramètre `--encryption-algorithm` est obligatoire. Comme dans toutes les commandes de la `encrypt` CLI, le `plaintext` paramètre doit être codé en base64, ou

vous devez utiliser le `fileb://` préfixe, qui indique à la AWS CLI de lire les données binaires du fichier.

```
aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --encryption-algorithm RSAES_OAEP_SHA_256 \
    --plaintext fileb://ExamplePlaintextFile \
    --output text \
    --query CiphertextBlob | base64 \
    --decode > ExampleEncryptedFile
```

Cette commande ne produit aucun résultat.

- Pour plus de détails sur l'API, voir Encrypt in AWS CLI Command Reference.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Encrypts the given text asynchronously using the specified KMS client and
key ID.
    *
    * @param keyId the ID of the KMS key to use for encryption
    * @param text the text to encrypt
    * @return a CompletableFuture that completes with the encrypted data as an
SdkBytes object
    */
   public CompletableFuture<SdkBytes> encryptDataAsync(String keyId, String
text) {
       SdkBytes myBytes = SdkBytes.fromUtf8String(text);
       EncryptRequest encryptRequest = EncryptRequest.builder()
           .keyId(keyId)
           .plaintext(myBytes)
           .build();
```

```
        CompletableFuture<EncryptResponse> responseFuture =
getAsyncClient().encrypt(encryptRequest).toCompletableFuture();
        return responseFuture.whenComplete((response, ex) -> {
            if (response != null) {
                String algorithm = response.encryptionAlgorithm().toString();
                logger.info("The string was encrypted with algorithm {}.",
algorithm);
            } else {
                throw new RuntimeException(ex);
            }
        }).thenApply(EncryptResponse::ciphertextBlob);
    }
```

- Pour plus de détails sur l'API, voir Encrypt in AWS SDK for Java 2.x API Reference.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun encryptData(keyIdValue: String): ByteArray? {
    val text = "This is the text to encrypt by using the AWS KMS Service"
    val myBytes: ByteArray = text.toByteArray()

    val encryptRequest =
        EncryptRequest {
            keyId = keyIdValue
            plaintext = myBytes
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.encrypt(encryptRequest)
        val algorithm: String = response.encryptionAlgorithm.toString()
        println("The encryption algorithm is $algorithm")
```

```
        // Return the encrypted data.
        return response.ciphertextBlob
    }
}

suspend fun decryptData(
    encryptedDataVal: ByteArray?,
    keyIdVal: String?,
) {
    val decryptRequest =
        DecryptRequest {
            ciphertextBlob = encryptedDataVal
            keyId = keyIdVal
        }
    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val decryptResponse = kmsClient.decrypt(decryptRequest)
        val myVal = decryptResponse.plaintext

        // Print the decrypted data.
        print(myVal)
    }
}
```

- Pour plus de détails sur l'API, voir Encrypt in AWS SDK for Kotlin API reference.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param string $text
     * @return Result
     */
```

```
    public function encrypt(string $keyId, string $text)
    {
        try {
            return $this->client->encrypt([
                'KeyId' => $keyId,
                'Plaintext' => $text,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "DisabledException"){
                echo "The request was rejected because the specified KMS key is
 not enabled.\n";
            }
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, voir Encrypt in AWS SDK pour PHP API Reference.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
 client.
        """
```

```
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def encrypt(self, key_id: str, text: str) -> str:
        """
        Encrypts text by using the specified key.

        :param key_id: The ARN or ID of the key to use for encryption.
        :param text: The text to encrypt.
        :return: The encrypted version of the text.
        """
        try:
            response = self.kms_client.encrypt(KeyId=key_id,
Plaintext=text.encode())
            print(
                f"The string was encrypted with algorithm
{response['EncryptionAlgorithm']}"
            )
            return response["CiphertextBlob"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DisabledException":
                logger.error(
                    "Could not encrypt because the key %s is disabled.", key_id
                )
            else:
                logger.error(
                    "Couldn't encrypt text. Here's why: %s",
                    err.response["Error"]["Message"],
                )
            raise
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API Encrypt in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'

# ARN of the AWS KMS key.
#
# Replace the fictitious key ARN with a valid key ID

keyId = 'arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab'

text = '1234567890'

client = Aws::KMS::Client.new(region: 'us-west-2')

resp = client.encrypt({
                        key_id: keyId,
                        plaintext: text
                      })

# Display a readable version of the resulting encrypted blob.
puts 'Blob:'
puts resp.ciphertext_blob.unpack('H*')
```

- Pour plus de détails sur l'API, voir Encrypt in AWS SDK pour Ruby API Reference.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn encrypt_string(
    verbose: bool,
    client: &Client,
    text: &str,
    key: &str,
    out_file: &str,
) -> Result<(), Error> {
    let blob = Blob::new(text.as_bytes());

    let resp = client.encrypt().key_id(key).plaintext(blob).send().await?;

    // Did we get an encrypted blob?
    let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);

    let mut ofile = File::create(out_file).expect("unable to create file");
    ofile.write_all(s.as_bytes()).expect("unable to write");

    if verbose {
        println!("Wrote the following to {:?}", out_file);
        println!("{}", s);
    }

    Ok(())
}
```

- Pour plus de détails sur l'API, voir Encrypt in AWS SDK for Rust API reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **GenerateDataKey** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GenerateDataKey`.

CLI

AWS CLI

Exemple 1 : pour générer une clé de données symétrique de 256 bits

L'`generate-data-key`exemple suivant demande une clé de données symétrique de 256 bits à utiliser en dehors de. AWS La commande renvoie une clé de données en texte brut pour une utilisation et une suppression immédiates, ainsi qu'une copie de cette clé de données chiffrée sous la clé KMS spécifiée. Vous pouvez stocker en toute sécurité la clé de données chiffrée avec les données chiffrées.

Pour demander une clé de données de 256 bits, utilisez le `key-spec` paramètre avec une valeur de. `AES_256` Pour demander une clé de données de 128 bits, utilisez le `key-spec` paramètre avec une valeur de. `AES_128` Pour toutes les autres longueurs de clé de données, utilisez le `number-of-bytes` paramètre.

La clé KMS que vous spécifiez doit être une clé KMS de chiffrement symétrique, c'est-à-dire une clé KMS dont la valeur de spécification de clé est SYMMETRIC_DEFAULT.

```
aws kms generate-data-key \
    --key-id alias/ExampleAlias \
    --key-spec AES_256
```

Sortie :

```
{
    "Plaintext": "VdzKNHGzUAzJeRBVY+uUmofUGGiDzyB3+i9fVkh3piw=",
    "KeyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
```

```
    "CiphertextBlob":
 "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIhvcNAQcGoG8wbQIBADBoBgk
+YdhV8MrkBQPeac0ReRVNDt9qleAt+SHgIRF8P0H+7U="
}
```

La `Plaintext` (clé de données en texte brut) et la `CiphertextBlob` (clé de données cryptée) sont renvoyées au format codé en base64.

Pour plus d'informations, consultez la section Clés de données dans le Guide du développeur du service de gestion des AWS clés. Exemple 2 : pour générer une clé de données symétrique 512 bits

L'`generate-data-key`exemple suivant demande une clé de données symétrique de 512 bits pour le chiffrement et le déchiffrement. La commande renvoie une clé de données en texte brut pour une utilisation et une suppression immédiates, ainsi qu'une copie de cette clé de données chiffrée sous la clé KMS spécifiée. Vous pouvez stocker en toute sécurité la clé de données chiffrée avec les données chiffrées.

Pour demander une longueur de clé autre que 128 ou 256 bits, utilisez le `number-of-bytes` paramètre. Pour demander une clé de données 512 bits, l'exemple suivant utilise le `number-of-bytes` paramètre avec une valeur de 64 (octets).

La clé KMS que vous spécifiez doit être une clé KMS de chiffrement symétrique, c'est-à-dire une clé KMS dont la valeur de spécification de clé est SYMMETRIC_DEFAULT.

REMARQUE : Les valeurs de la sortie de cet exemple sont tronquées pour être affichées.

```
aws kms generate-data-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --number-of-bytes 64
```

Sortie :

```
{
    "CiphertextBlob": "AQIBAHi6LtupRpdKl2aJTzkK6FbhOtQkMlQJJH3PdtHvS/y+hAEnX/
QQNmMwDfg2korNMEc8AAACaDCCAmQGCSqGSIb3DQEHBqCCAlUwggJRAgEAMIICSgYJKoZ...",
    "Plaintext": "ty8Lr0Bk6OF07M2BWt6qbFdNB
+G00ZLtf5MSEb4al3R2UKWGOp06njAwy2n72VRm2m7z/
Pm9Wpbvttz6a4lSo9hgPvKhZ5y6RTm4OovEXiVfBveyX3DQxDzRSwbKDPk/...",
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

Les `Plaintext` (clé de données en texte brut) et `CiphertextBlob` (clé de données cryptée) sont renvoyées au format codé en base64.

Pour plus d'informations, consultez la section Clés de données dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous GenerateDataKeyà la section Référence des AWS CLI commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def generate_data_key(self, key_id):
        """
```

```
        Generates a symmetric data key that can be used for client-side
 encryption.
        """
        answer = input(
            f"Do you want to generate a symmetric data key from key {key_id} (y/
n)? "
        )
        if answer.lower() == "y":
            try:
                data_key = self.kms_client.generate_data_key(
                    KeyId=key_id, KeySpec="AES_256"
                )
            except ClientError as err:
                logger.error(
                    "Couldn't generate a data key for key %s. Here's why: %s",
                    key_id,
                    err.response["Error"]["Message"],
                )
            else:
                pprint(data_key)
```

- Pour plus de détails sur l'API, consultez GenerateDataKeyle AWS manuel de référence de l'API SDK for Python (Boto3).

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn make_key(client: &Client, key: &str) -> Result<(), Error> {
    let resp = client
        .generate_data_key()
        .key_id(key)
        .key_spec(DataKeySpec::Aes256)
```

```
        .send()
        .await?;

    // Did we get an encrypted blob?
    let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);

    println!();
    println!("Data key:");
    println!("{}", s);

    Ok(())
}
```

- Pour plus de détails sur l'API, voir GenerateDataKeyla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GenerateDataKeyWithoutPlaintext`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GenerateDataKeyWithoutPlaintext`.

CLI

AWS CLI

Pour générer une clé de données symétrique de 256 bits sans clé en texte brut

L'`generate-data-key-without-plaintext`exemple suivant demande une copie cryptée d'une clé de données symétrique de 256 bits pour une utilisation en dehors de. AWS Vous pouvez appeler AWS KMS pour déchiffrer la clé de données lorsque vous êtes prêt à l'utiliser.

Pour demander une clé de données de 256 bits, utilisez le `key-spec` paramètre avec une valeur de. `AES_256` Pour demander une clé de données de 128 bits, utilisez le `key-spec` paramètre avec une valeur de. `AES_128` Pour toutes les autres longueurs de clé de données, utilisez le `number-of-bytes` paramètre.

La clé KMS que vous spécifiez doit être une clé KMS de chiffrement symétrique, c'est-à-dire une clé KMS dont la valeur de spécification de clé est SYMMETRIC_DEFAULT.

```
aws kms generate-data-key-without-plaintext \
    --key-id "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab" \
    --key-spec AES_256
```

Sortie :

```
{
    "CiphertextBlob":
 "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIhvcNAQcGoG8wbQIBADBoBgk
    "KeyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
 "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

La `CiphertextBlob` (clé de données cryptée) est renvoyée au format codé en base64.

Pour plus d'informations, consultez la section Clés de données dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous GenerateDataKeyWithoutPlaintextà la section Référence des AWS CLI commandes.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn make_key(client: &Client, key: &str) -> Result<(), Error> {
    let resp = client
        .generate_data_key_without_plaintext()
```

```
        .key_id(key)
        .key_spec(DataKeySpec::Aes256)
        .send()
        .await?;

    // Did we get an encrypted blob?
    let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);

    println!();
    println!("Data key:");
    println!("{}", s);

    Ok(())
}
```

- Pour plus de détails sur l'API, voir GenerateDataKeyWithoutPlaintextla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GenerateRandom`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GenerateRandom`.

CLI

AWS CLI

Exemple 1 : pour générer une chaîne d'octets aléatoire de 256 bits (Linux ou macOS)

L'`generate-random`exemple suivant génère une chaîne d'octets aléatoire de 256 bits (32 octets) codée en base64. L'exemple décode la chaîne d'octets et l'enregistre dans le fichier aléatoire.

Lorsque vous exécutez cette commande, vous devez utiliser le `number-of-bytes` paramètre pour spécifier la longueur de la valeur aléatoire en octets.

Vous ne spécifiez pas de clé KMS lorsque vous exécutez cette commande. La chaîne d'octets aléatoire n'est liée à aucune clé KMS.

Par défaut, AWS KMS génère le nombre aléatoire. Toutefois, si vous spécifiez un magasin de clés personnalisé, la chaîne d'octets aléatoire est générée dans le cluster AWS CloudHSM associé au magasin de clés personnalisé.

Cet exemple utilise les paramètres et valeurs suivants :

Il utilise le `--number-of-bytes` paramètre requis avec une valeur de `32` pour demander une chaîne de 32 octets (256 bits). Il utilise le `--output` paramètre avec une valeur de pour demander à la AWS CLI de renvoyer la sortie sous forme de texte, au lieu de JSON. Il utilise le `--query parameter` pour extraire la valeur de la `Plaintext` propriété de la réponse. Il envoie (|) la sortie de la commande à l'`base64`utilitaire, qui décode la sortie extraite. Il utilise l'opérateur de `text` redirection (**>**) pour enregistrer la chaîne d'octets décodée dans le fichier .it utilise l'opérateur de redirection (**>**) `ExampleRandom` pour enregistrer le texte chiffré binaire dans un fichier.

```
aws kms generate-random \
    --number-of-bytes 32 \
    --output text \
    --query Plaintext | base64 --decode > ExampleRandom
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la référence GenerateRandomde l'API du service de gestion des AWS clés.

Exemple 2 : pour générer un nombre aléatoire de 256 bits (invite de commande Windows)

L'exemple suivant utilise la `generate-random` commande pour générer une chaîne d'octets aléatoire de 256 bits (32 octets) codée en base64. L'exemple décode la chaîne d'octets et l'enregistre dans le fichier aléatoire. Cet exemple est identique à l'exemple précédent, sauf qu'il utilise l'`certutil`utilitaire de Windows pour décoder en base64 la chaîne d'octets aléatoire avant de l'enregistrer dans un fichier.

Tout d'abord, générez une chaîne d'octets aléatoire codée en base64 et enregistrez-la dans un fichier temporaire. `ExampleRandom.base64`

```
aws kms generate-random \
    --number-of-bytes 32 \
```

```
    --output text \
    --query Plaintext > ExampleRandom.base64
```

La sortie de la `generate-random` commande étant enregistrée dans un fichier, cet exemple ne produit aucune sortie.

Utilisez maintenant la `certutil -decode` commande pour décoder la chaîne d'octets codée en base64 dans le fichier. `ExampleRandom.base64` Ensuite, il enregistre la chaîne d'octets décodée dans le `ExampleRandom` fichier.

```
certutil -decode ExampleRandom.base64 ExampleRandom
```

Sortie :

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

Pour plus d'informations, consultez la référence GenerateRandomde l'API du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous GenerateRandomà la section Référence des AWS CLI commandes.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn make_string(client: &Client, length: i32) -> Result<(), Error> {
    let resp = client
        .generate_random()
        .number_of_bytes(length)
        .send()
        .await?;
```

```
    // Did we get an encrypted blob?
    let blob = resp.plaintext.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);

    println!();
    println!("Data key:");
    println!("{}", s);

    Ok(())
}
```

- Pour plus de détails sur l'API, voir GenerateRandomla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetKeyPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetKeyPolicy`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour copier une politique clé d'une clé KMS vers une autre clé KMS

L'`get-key-policy`exemple suivant extrait la politique clé d'une clé KMS et l'enregistre dans un fichier texte. Il remplace ensuite la politique d'une autre clé KMS en utilisant le fichier texte comme entrée de politique.

Comme le `--policy` paramètre de `put-key-policy` nécessite une chaîne, vous devez utiliser l'`--output text`option pour renvoyer la sortie sous forme de chaîne de texte au lieu de JSON.

```
aws kms get-key-policy \
    --policy-name default \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --query Policy \
    --output text > policy.txt

aws kms put-key-policy \
    --policy-name default \
    --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --policy file://policy.txt
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez le PutKeyPolicymanuel de référence de l'API AWS KMS.

- Pour plus de détails sur l'API, reportez-vous GetKeyPolicyà la section Référence des AWS CLI commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.
```

```
        :return: An instance of KeyPolicy initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def get_policy(self, key_id: str) -> dict[str, str]:
        """
        Gets the policy of a key.

        :param key_id: The ARN or ID of the key to query.
        :return: The key policy as a dict.
        """
        if key_id != "":
            try:
                response = self.kms_client.get_key_policy(
                    KeyId=key_id,
                )
                policy = json.loads(response["Policy"])
            except ClientError as err:
                logger.error(
                    "Couldn't get policy for key %s. Here's why: %s",
                    key_id,
                    err.response["Error"]["Message"],
                )
                raise
            else:
                pprint(policy)
                return policy
        else:
            print("Skipping get policy demo.")
```

- Pour plus de détails sur l'API, consultez GetKeyPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListAliases`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ListAliases`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.KeyManagementService;
    using Amazon.KeyManagementService.Model;

    /// <summary>
    /// List the AWS Key Management Service (AWS KMS) aliases that have been
defined for
    /// the keys in the same AWS Region as the default user. If you want to list
    /// the aliases in a different Region, pass the Region to the client
    /// constructor.
    /// </summary>
    public class ListAliases
    {
        public static async Task Main()
        {
            var client = new AmazonKeyManagementServiceClient();
            var request = new ListAliasesRequest();
            var response = new ListAliasesResponse();

            do
```

```
            {
                response = await client.ListAliasesAsync(request);

                response.Aliases.ForEach(alias =>
                {
                    Console.WriteLine($"Created: {alias.CreationDate} Last
 Update: {alias.LastUpdatedDate} Name: {alias.AliasName}");
                });

                request.Marker = response.NextMarker;
            }
            while (response.Truncated);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAliasesà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Exemple 1 : pour répertorier tous les alias d'un AWS compte et d'une région

L'exemple suivant utilise la `list-aliases` commande pour répertorier tous les alias de la région par défaut du AWS compte. La sortie inclut les alias associés aux clés KMS AWS gérées et aux clés KMS gérées par le client.

```
aws kms list-aliases
```

Sortie :

```
{
    "Aliases": [
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/testKey",
            "AliasName": "alias/testKey",
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
```

```
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/FinanceDept",
            "AliasName": "alias/FinanceDept",
            "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
        },
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/dynamodb",
            "AliasName": "alias/aws/dynamodb",
            "TargetKeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
        },
        {
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/ebs",
            "AliasName": "alias/aws/ebs",
            "TargetKeyId": "0987ab65-43cd-21ef-09ab-87654321cdef"
        },
        ...
    ]
}
```

Exemple 2 : pour répertorier tous les alias d'une clé KMS particulière

L'exemple suivant utilise la `list-aliases` commande et son `key-id` paramètre pour répertorier tous les alias associés à une clé KMS particulière.

Chaque alias est associé à une seule clé KMS, mais une clé KMS peut avoir plusieurs alias. Cette commande est très utile car la console AWS KMS ne répertorie qu'un seul alias pour chaque clé KMS. Pour trouver tous les alias d'une clé KMS, vous devez utiliser la `list-aliases` commande.

Cet exemple utilise l'ID de clé KMS pour le `--key-id` paramètre, mais vous pouvez utiliser un ID de clé, un ARN de clé, un nom d'alias ou un ARN d'alias dans cette commande.

```
aws kms list-aliases --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Sortie :

```
{
    "Aliases": [
        {
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/oregon-test-
key",
            "AliasName": "alias/oregon-test-key"
```

```
        },
        {
            "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
            "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/project121-
test",
            "AliasName": "alias/project121-test"
        }
    ]
}
```

Pour plus d'informations, consultez la section Travailler avec des alias dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous ListAliasesà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Asynchronously lists all the aliases in the current AWS account.
     *
     * @return a {@link CompletableFuture} that completes when the list of
aliases has been processed
     */
    public CompletableFuture<Object> listAllAliasesAsync() {
        ListAliasesRequest aliasesRequest = ListAliasesRequest.builder()
            .limit(15)
            .build();

        ListAliasesPublisher paginator =
getAsyncClient().listAliasesPaginator(aliasesRequest);
        return paginator.subscribe(response -> {
                response.aliases().forEach(alias ->
                    logger.info("The alias name is: " + alias.aliasName())
```

```
                );
            })
            .thenApply(v -> null)
            .exceptionally(ex -> {
                if (ex.getCause() instanceof KmsException) {
                    KmsException e = (KmsException) ex.getCause();
                    throw new RuntimeException("A KMS exception occurred: " +
 e.getMessage());
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
 ex.getMessage());
                }
            });
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAliasesà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listAllAliases() {
    val request =
        ListAliasesRequest {
            limit = 15
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.listAliases(request)
        response.aliases?.forEach { alias ->
            println("The alias name is ${alias.aliasName}")
        }
    }
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous ListAliasesà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param int $limit
     * @return ResultPaginator
     */
    public function listAliases(string $keyId = "", int $limit = 0)
    {
        $args = [];
        if($keyId){
            $args['KeyId'] = $keyId;
        }
        if($limit){
            $args['Limit'] = $limit;
        }
        try{
            return $this->client->getPaginator("ListAliases", $args);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidMarkerException"){
                echo "The request was rejected because the marker that specifies
where pagination should next begin is not valid.\n";
            }
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAliasesà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def list_aliases(self, page_size: int) -> None:
        """
        Lists aliases for the current account.
        :param page_size: The number of aliases to list per page.
        """
        try:
            alias_paginator = self.kms_client.get_paginator("list_aliases")
            for alias_page in alias_paginator.paginate(
```

```
                PaginationConfig={"PageSize": page_size}
            ):
                print(f"Here are {page_size} aliases:")
                pprint(alias_page["Aliases"])
                if alias_page["Truncated"]:
                    answer = input(
                        f"Do you want to see the next {page_size} aliases (y/n)?
"
                    )
                    if answer.lower() != "y":
                        break
                else:
                    print("That's all your aliases!")
        except ClientError as err:
            logging.error(
                "Couldn't list your aliases. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez ListAliasesle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListGrants`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ListGrants`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   using System;
   using System.Threading.Tasks;
   using Amazon.KeyManagementService;
   using Amazon.KeyManagementService.Model;

   /// <summary>
   /// List the AWS Key Management Service (AWS KMS) grants that are associated
with
   /// a specific key.
   /// </summary>
   public class ListGrants
   {
       public static async Task Main()
       {
           // The identifier of the AWS KMS key to disable. You can use the
           // key Id or the Amazon Resource Name (ARN) of the AWS KMS key.
           var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";
           var client = new AmazonKeyManagementServiceClient();
           var request = new ListGrantsRequest
           {
               KeyId = keyId,
           };

           var response = new ListGrantsResponse();

           do
           {
               response = await client.ListGrantsAsync(request);

               response.Grants.ForEach(grant =>
               {
                   Console.WriteLine($"{grant.GrantId}");
```

```
                });

                request.Marker = response.NextMarker;
            }
            while (response.Truncated);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGrantsà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour afficher les subventions sur une clé AWS KMS

L'`list-grants`exemple suivant affiche toutes les autorisations associées à la clé KMS AWS gérée spécifiée pour Amazon DynamoDB dans votre compte. Cette autorisation permet à DynamoDB d'utiliser la clé KMS en votre nom pour chiffrer une table DynamoDB avant de l'écrire sur le disque. Vous pouvez utiliser une commande comme celle-ci pour afficher les autorisations relatives aux clés KMS AWS gérées et aux clés KMS gérées par le client dans le AWS compte et la région.

Cette commande utilise le `key-id` paramètre avec un ID de clé pour identifier la clé KMS. Vous pouvez utiliser un ID de clé ou un ARN de clé pour identifier la clé KMS. Pour obtenir l'ID de clé ou l'ARN d'une clé KMS AWS gérée, utilisez la `list-aliases` commande `list-keys` or.

```
aws kms list-grants \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Le résultat indique que l'autorisation autorise Amazon DynamoDB à utiliser la clé KMS pour des opérations cryptographiques, à consulter les informations relatives à la clé KMS `DescribeKey` () et à retirer les autorisations (). `RetireGrant` La `EncryptionContextSubset` contrainte limite ces autorisations aux demandes qui incluent les paires de contextes de chiffrement spécifiées. Par conséquent, les autorisations de la subvention ne sont effectives que sur le compte spécifié et sur la table DynamoDB.

```
{
    "Grants": [
        {
            "Constraints": {
                "EncryptionContextSubset": {
                    "aws:dynamodb:subscriberId": "123456789012",
                    "aws:dynamodb:tableName": "Services"
                }
            },
            "IssuingAccount": "arn:aws:iam::123456789012:root",
            "Name": "8276b9a6-6cf0-46f1-b2f0-7993a7f8c89a",
            "Operations": [
                "Decrypt",
                "Encrypt",
                "GenerateDataKey",
                "ReEncryptFrom",
                "ReEncryptTo",
                "RetireGrant",
                "DescribeKey"
            ],
            "GrantId":
 "1667b97d27cf748cf05b487217dd4179526c949d14fb3903858e25193253fe59",
            "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "RetiringPrincipal": "dynamodb.us-west-2.amazonaws.com",
            "GranteePrincipal": "dynamodb.us-west-2.amazonaws.com",
            "CreationDate": "2021-05-13T18:32:45.144000+00:00"
        }
    ]
}
```

Pour plus d'informations, consultez la section Subventions dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous ListGrantsà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Asynchronously displays the grant IDs for the specified key ID.
     *
     * @param keyId the ID of the AWS KMS key for which to list the grants
     * @return a {@link CompletableFuture} that, when completed, will be null
if the operation succeeded, or will throw a {@link RuntimeException} if the
operation failed
     * @throws RuntimeException if there was an error listing the grants, either
due to an {@link KmsException} or an unexpected error
     */
    public CompletableFuture<Object> displayGrantIdsAsync(String keyId) {
        ListGrantsRequest grantsRequest = ListGrantsRequest.builder()
            .keyId(keyId)
            .limit(15)
            .build();

        ListGrantsPublisher paginator =
getAsyncClient().listGrantsPaginator(grantsRequest);
        return paginator.subscribe(response -> {
                response.grants().forEach(grant -> {
                    logger.info("The grant Id is: " + grant.grantId());
                });
            })
            .thenApply(v -> null)
            .exceptionally(ex -> {
                Throwable cause = ex.getCause();
                if (cause instanceof KmsException) {
                    throw new RuntimeException("Failed to list grants: " +
cause.getMessage(), cause);
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
cause.getMessage(), cause);
```

```
                }
            });
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGrantsà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun displayGrantIds(keyIdVal: String?) {
    val request =
        ListGrantsRequest {
            keyId = keyIdVal
            limit = 15
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.listGrants(request)
        response.grants?.forEach { grant ->
            println("The grant Id is ${grant.grantId}")
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListGrantsà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @return Result
     */
    public function listGrants(string $keyId)
    {
        try{
            return $this->client->listGrants([
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "    The request was rejected because the specified entity
or resource could not be found.\n";
            }
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGrantsà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def list_grants(self, key_id):
        """
        Lists grants for a key.

        :param key_id: The ARN or ID of the key to query.
        :return: The grants for the key.
        """
        try:
            paginator = self.kms_client.get_paginator("list_grants")
            grants = []
            page_iterator = paginator.paginate(KeyId=key_id)
            for page in page_iterator:
                grants.extend(page["Grants"])

            print(f"Grants for key {key_id}:")
            pprint(grants)
```

```
            return grants
        except ClientError as err:
            logger.error(
                "Couldn't list grants for key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez ListGrantsle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListKeyPolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ListKeyPolicies`.

CLI

AWS CLI

Pour obtenir les noms des politiques clés pour une clé KMS

L'`list-key-policies`exemple suivant obtient les noms des politiques clés pour une clé gérée par le client dans les exemples de compte et de région. Vous pouvez utiliser cette commande pour rechercher les noms des politiques clés relatives aux clés AWS gérées et aux clés gérées par le client.

Comme le seul nom de politique clé valide est`default`, cette commande n'est pas utile.

Pour spécifier la clé KMS, utilisez le `key-id` paramètre. Cet exemple utilise une valeur d'ID de clé, mais vous pouvez utiliser un ID de clé ou un ARN de clé dans cette commande.

```
aws kms list-key-policies \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Sortie :

```
{
    "PolicyNames": [
    "default"
    ]
}
```

Pour plus d'informations sur les politiques clés AWS KMS, consultez la section Utilisation des politiques clés dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous ListKeyPoliciesà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Asynchronously retrieves the key policy for the specified key ID and
policy name.
    *
    * @param keyId       the ID of the AWS KMS key for which to retrieve the
policy
    * @param policyName the name of the key policy to retrieve
    * @return a {@link CompletableFuture} that, when completed, contains the key
policy as a {@link String}
    */
   public CompletableFuture<String> getKeyPolicyAsync(String keyId, String
policyName) {
       GetKeyPolicyRequest policyRequest = GetKeyPolicyRequest.builder()
           .keyId(keyId)
           .policyName(policyName)
           .build();

       return getAsyncClient().getKeyPolicy(policyRequest)
```

```
            .thenApply(response -> {
                String policy = response.policy();
                logger.info("The response is: " + policy);
                return policy;
            })
            .exceptionally(ex -> {
                throw new RuntimeException("Failed to get key policy", ex);
            });
    }
```

- Pour plus de détails sur l'API, reportez-vous ListKeyPolicieslà la section Référence des AWS SDK for Java 2.x API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.

        :return: An instance of KeyPolicy initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def list_policies(self, key_id):
```

```
        """
        Lists the names of the policies for a key.

        :param key_id: The ARN or ID of the key to query.
        """
        try:
            policy_names = self.kms_client.list_key_policies(KeyId=key_id)[
                "PolicyNames"
            ]
        except ClientError as err:
            logging.error(
                "Couldn't list your policies. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
        else:
            print(f"The policies for key {key_id} are:")
            pprint(policy_names)
```

- Pour plus de détails sur l'API, consultez ListKeyPolicieslle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListKeys`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ListKeys`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   using System;
   using System.Threading.Tasks;
   using Amazon.KeyManagementService;
   using Amazon.KeyManagementService.Model;

   /// <summary>
   /// List the AWS Key Managements Service (AWS KMS) keys for the AWS Region
   /// of the default user. To list keys in another AWS Region, supply the
Region
   /// as a parameter to the client constructor.
   /// </summary>
   public class ListKeys
   {
       public static async Task Main()
       {
           var client = new AmazonKeyManagementServiceClient();
           var request = new ListKeysRequest();
           var response = new ListKeysResponse();

           do
           {
               response = await client.ListKeysAsync(request);

               response.Keys.ForEach(key =>
               {
                   Console.WriteLine($"ID: {key.KeyId}, {key.KeyArn}");
               });

               // Set the Marker property when response.Truncated is true
               // in order to get the next keys.
               request.Marker = response.NextMarker;
           }
```

```
            while (response.Truncated);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK pour .NET API.

CLI

AWS CLI

Pour obtenir les clés KMS dans un compte et une région

L'`list-keys`exemple suivant permet d'obtenir les clés KMS d'un compte et d'une région. Cette commande renvoie à la fois les clés AWS gérées et les clés gérées par le client.

```
aws kms list-keys
```

Sortie :

```
{
    "Keys": [
        {
            "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        {
            "KeyArn": "arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
            "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
        },
        {
            "KeyArn": "arn:aws:kms:us-
east-2:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d",
            "KeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
        }
    ]
}
```

Pour plus d'informations, consultez la section Visualisation des clés dans le guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.kms.KmsAsyncClient;
import software.amazon.awssdk.services.kms.model.ListKeysRequest;
import software.amazon.awssdk.services.kms.paginators.ListKeysPublisher;
import java.util.concurrent.CompletableFuture;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloKMS {
    public static void main(String[] args) {
        listAllKeys();
    }

    public static void listAllKeys() {
        KmsAsyncClient kmsAsyncClient = KmsAsyncClient.builder()
            .build();
        ListKeysRequest listKeysRequest = ListKeysRequest.builder()
            .limit(15)
            .build();
```

```
        /*
         * The `subscribe` method is required when using paginator methods in the
 AWS SDK
         * because paginator methods return an instance of a `ListKeysPublisher`,
 which is
         * based on a reactive stream. This allows asynchronous retrieval of
 paginated
         * results as they become available. By subscribing to the stream, we can
 process
         * each page of results as they are emitted.
         */
        ListKeysPublisher keysPublisher =
 kmsAsyncClient.listKeysPaginator(listKeysRequest);
        CompletableFuture<Void> future = keysPublisher
            .subscribe(r -> r.keys().forEach(key ->
                System.out.println("The key ARN is: " + key.keyArn() + ". The key
 Id is: " + key.keyId())))
            .whenComplete((result, exception) -> {
                if (exception != null) {
                    System.err.println("Error occurred: " +
 exception.getMessage());
                } else {
                    System.out.println("Successfully listed all keys.");
                }
            });

        try {
            future.join();
        } catch (Exception e) {
            System.err.println("Failed to list keys: " + e.getMessage());
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK for Java 2.x API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listAllKeys() {
    val request =
        ListKeysRequest {
            limit = 15
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.listKeys(request)
        response.keys?.forEach { key ->
            println("The key ARN is ${key.keyArn}")
            println("The key Id is ${key.keyId}")
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @return array
     */
    public function listKeys()
    {
        try {
            $contents = [];
            $paginator = $this->client->getPaginator("ListKeys");
            foreach($paginator as $result){
                foreach ($result['Content'] as $object) {
                    $contents[] = $object;
                }
            }
            return $contents;
        }catch(KmsException $caught){
            echo "There was a problem listing the keys: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ListKeysà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []
```

```
    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def list_keys(self):
        """
        Lists the keys for the current account by using a paginator.
        """
        try:
            page_size = 10
            print("\nLet's list your keys.")
            key_paginator = self.kms_client.get_paginator("list_keys")
            for key_page in key_paginator.paginate(PaginationConfig={"PageSize":
10}):
                print(f"Here are {len(key_page['Keys'])} keys:")
                pprint(key_page["Keys"])
                if key_page["Truncated"]:
                    answer = input(
                        f"Do you want to see the next {page_size} keys (y/n)? "
                    )
                    if answer.lower() != "y":
                        break
                else:
                    print("That's all your keys!")
        except ClientError as err:
            logging.error(
                "Couldn't list your keys. Here's why: %s",
                err.response["Error"]["Message"],
            )
```

- Pour plus de détails sur l'API, consultez ListKeysle AWS manuel de référence de l'API SDK for Python (Boto3).

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn show_keys(client: &Client) -> Result<(), Error> {
    let resp = client.list_keys().send().await?;

    let keys = resp.keys.unwrap_or_default();

    let len = keys.len();

    for key in keys {
        println!("Key ARN: {}", key.key_arn.as_deref().unwrap_or_default());
    }

    println!();
    println!("Found {} keys", len);

    Ok(())
}
```

- Pour plus de détails sur l'API, voir ListKeysla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`PutKeyPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `PutKeyPolicy`.

CLI

AWS CLI

Pour modifier la politique de clé d'une clé KMS

L'`put-key-policy`exemple suivant modifie la politique de clé pour une clé gérée par le client.

Pour commencer, créez une politique clé et enregistrez-la dans un fichier JSON local. Dans cet exemple, le fichier est`key_policy.json`. Vous pouvez également spécifier la politique clé sous forme de valeur de chaîne du `policy` paramètre.

La première déclaration de cette politique clé autorise le AWS compte à utiliser les politiques IAM pour contrôler l'accès à la clé KMS. La deuxième instruction autorise l'`test-user`utilisateur à exécuter les `list-keys` commandes `describe-key` et sur la clé KMS.

Contenu de `key_policy.json` :

```
{
    "Version" : "2012-10-17",
    "Id" : "key-default-1",
    "Statement" : [
        {
            "Sid" : "Enable IAM User Permissions",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:root"
            },
            "Action" : "kms:*",
            "Resource" : "*"
        },
        {
            "Sid" : "Allow Use of Key",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:user/test-user"
            },
            "Action" : [
                "kms:DescribeKey",
                "kms:ListKeys"
            ],
            "Resource" : "*"
```

```
        }
    ]
}
```

Pour identifier la clé KMS, cet exemple utilise l'ID de la clé, mais vous pouvez également utiliser un ARN de clé. Pour spécifier la politique clé, la commande utilise le `policy` paramètre. Pour indiquer que la politique se trouve dans un fichier, elle utilise le `file://` préfixe requis. Ce préfixe est nécessaire pour identifier les fichiers sur tous les systèmes d'exploitation pris en charge. Enfin, la commande utilise le `policy-name` paramètre avec une valeur de`default`. Si aucun nom de politique n'est spécifié, la valeur par défaut est`default`. La seule valeur valide est `default`.

```
aws kms put-key-policy \
    --policy-name default \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --policy file://key_policy.json
```

Cette commande ne génère pas de sortie. Pour vérifier que la commande est efficace, `get-key-policy` utilisez-la. L'exemple de commande suivant permet d'obtenir la politique de clé pour la même clé KMS. Le `output` paramètre avec une valeur de `text` renvoie un format de texte facile à lire.

```
aws kms get-key-policy \
    --policy-name default \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --output text
```

Sortie :

```
{
    "Version" : "2012-10-17",
    "Id" : "key-default-1",
    "Statement" : [
        {
            "Sid" : "Enable IAM User Permissions",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:root"
            },
            "Action" : "kms:*",
```

```
            "Resource" : "*"
            },
            {
            "Sid" : "Allow Use of Key",
            "Effect" : "Allow",
            "Principal" : {
                "AWS" : "arn:aws:iam::111122223333:user/test-user"
            },
            "Action" : [ "kms:Describe", "kms:List" ],
            "Resource" : "*"
        }
    ]
}
```

Pour plus d'informations, consultez la section Modification d'une politique clé dans le guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous PutKeyPolicyà la section Référence des AWS CLI commandes.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param string $policy
     * @return void
     */
    public function putKeyPolicy(string $keyId, string $policy)
    {
        try {
            $this->client->putKeyPolicy([
                'KeyId' => $keyId,
                'Policy' => $policy,
```

```
            ]);
        }catch(KmsException $caught){
            echo "There was a problem replacing the key policy: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous PutKeyPolicyà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.

        :return: An instance of KeyPolicy initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def set_policy(self, key_id: str, policy: dict[str, any]) -> None:
        """
```

```
        Sets the policy of a key. Setting a policy entirely overwrites the
existing
        policy, so care is taken to add a statement to the existing list of
statements
        rather than simply writing a new policy.

        :param key_id: The ARN or ID of the key to set the policy to.
        :param policy: The existing policy of the key.
        :return: None
        """
        principal = input(
            "Enter the ARN of an IAM role to set as the principal on the policy:
"
        )
        if key_id != "" and principal != "":
            # The updated policy replaces the existing policy. Add a new
statement to
            # the list along with the original policy statements.
            policy["Statement"].append(
                {
                    "Sid": "Allow access for ExampleRole",
                    "Effect": "Allow",
                    "Principal": {"AWS": principal},
                    "Action": [
                        "kms:Encrypt",
                        "kms:GenerateDataKey*",
                        "kms:Decrypt",
                        "kms:DescribeKey",
                        "kms:ReEncrypt*",
                    ],
                    "Resource": "*",
                }
            )
            try:
                self.kms_client.put_key_policy(KeyId=key_id,
Policy=json.dumps(policy))
            except ClientError as err:
                logger.error(
                    "Couldn't set policy for key %s. Here's why %s",
                    key_id,
                    err.response["Error"]["Message"],
                )
                raise
            else:
```

```
                print(f"Set policy for key {key_id}.")
        else:
            print("Skipping set policy demo.")
```

- Pour plus de détails sur l'API, consultez PutKeyPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ReEncrypt`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ReEncrypt`.

CLI

AWS CLI

Exemple 1 : rechiffrer un message chiffré sous une autre clé KMS symétrique (Linux et macOS).

L'exemple de `re-encrypt` commande suivant illustre la méthode recommandée pour rechiffrer les données à l'aide de la CLI AWS .

Fournissez le texte chiffré dans un fichier. Dans la valeur du `--ciphertext-blob` paramètre, utilisez le `fileb://` préfixe, qui indique à la CLI de lire les données d'un fichier binaire. Si le fichier ne se trouve pas dans le répertoire actuel, saisissez le chemin complet du fichier. Pour plus d'informations sur la lecture des valeurs des paramètres de la AWS CLI à partir d'un fichier, consultez la section Chargement de paramètres AWS CLI depuis un fichier dans le guide de l'utilisateur de l'interface de ligne de AWS commande et les meilleures pratiques relatives aux paramètres de fichiers locaux dans le blog des outils de ligne de AWS commande. Spécifiez la clé KMS source, qui déchiffre le texte chiffré. Le `--source-key-id` paramètre n'est pas obligatoire lors du déchiffrement à l'aide de clés KMS de chiffrement symétriques. AWS KMS peut obtenir la clé KMS qui a été utilisée pour chiffrer les données à partir des métadonnées du blob de texte chiffré. Toutefois, la spécification de la clé KMS que vous utilisez est une bonne pratique. Cette pratique garantit que vous utilisez la clé KMS comme vous le souhaitez et vous empêche de déchiffrer par inadvertance un texte chiffré à

l'aide d'une clé KMS non fiable. Spécifiez la clé KMS de destination, qui chiffre à nouveau les données. Le paramètre est toujours obligatoire. `--destination-key-id` Cet exemple utilise un ARN de clé, mais vous pouvez utiliser n'importe quel identifiant de clé valide. Demandez la sortie en texte brut sous forme de valeur de texte.Le `--query` paramètre indique à la CLI de n'obtenir que la valeur du champ à partir de `Plaintext` la sortie. Le `--output` paramètre renvoie la sortie sous forme de texte.Base64 décodez le texte en clair et enregistrez-le dans un fichier. L'exemple suivant montre comment rediriger (|) la valeur du `Plaintext` paramètre vers l'utilitaire Base64, qui le décode. Ensuite, il redirige (>) la sortie décodée vers le `ExamplePlaintext` fichier.

Avant d'exécuter cette commande, remplacez l'exemple de clé par IDs des identifiants de clé valides provenant de votre AWS compte.

```
aws kms re-encrypt \
    --ciphertext-blob fileb://ExampleEncryptedFile \
    --source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
    --query CiphertextBlob \
    --output text | base64 --decode > ExampleReEncryptedFile
```

Cette commande ne produit aucun résultat. La sortie de la `re-encrypt` commande est décodée en base64 et enregistrée dans un fichier.

Pour plus d'informations, consultez la référence ReEncryptde l'API du service de gestion des AWS clés.

Exemple 2 : pour rechiffrer un message chiffré sous une autre clé KMS symétrique (invite de commande Windows).

L'exemple de `re-encrypt` commande suivant est identique au précédent, sauf qu'il utilise l'`certutil`utilitaire pour décoder les données en texte brut en Base64. Cette procédure nécessite deux commandes, comme indiqué dans les exemples suivants.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte.

```
aws kms re-encrypt ^
    --ciphertext-blob fileb://ExampleEncryptedFile ^
    --source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^
    --destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 ^
```

```
    --query CiphertextBlob ^
    --output text > ExampleReEncryptedFile.base64
```

Ensuite, utilisez l'`certutil`utilitaire

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

Sortie :

```
Input Length = 18
Output Length = 12
CertUtil: -decode command completed successfully.
```

Pour plus d'informations, consultez la référence ReEncryptde l'API du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous ReEncryptà la section Référence des AWS CLI commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
 client.
```

```
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def re_encrypt(self, source_key_id, cipher_text):
        """
        Takes ciphertext previously encrypted with one key and reencrypt it by
using
        another key.

        :param source_key_id: The ARN or ID of the original key used to encrypt
the
                              ciphertext.
        :param cipher_text: The encrypted ciphertext.
        :return: The ciphertext encrypted by the second key.
        """
        destination_key_id = input(
            f"Your ciphertext is currently encrypted with key {source_key_id}. "
            f"Enter another key ID or ARN to reencrypt it: "
        )
        if destination_key_id != "":
            try:
                cipher_text = self.kms_client.re_encrypt(
                    SourceKeyId=source_key_id,
                    DestinationKeyId=destination_key_id,
                    CiphertextBlob=cipher_text,
                )["CiphertextBlob"]
            except ClientError as err:
                logger.error(
                    "Couldn't reencrypt your ciphertext. Here's why: %s",
                    err.response["Error"]["Message"],
                )
            else:
                print(f"Reencrypted your ciphertext as: {cipher_text}")
                return cipher_text
        else:
            print("Skipping reencryption demo.")
```

- Pour plus de détails sur l'API, consultez ReEncryptle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'

# Human-readable version of the ciphertext of the data to reencrypt.

blob =
 '01020200785d68faeec386af1057904926253051eb2919d3c16078badf65b808b26dd057c101747cadf3593
sourceCiphertextBlob = [blob].pack('H*')

# Replace the fictitious key ARN with a valid key ID

destinationKeyId = 'arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321'

client = Aws::KMS::Client.new(region: 'us-west-2')

resp = client.re_encrypt({
                           ciphertext_blob: sourceCiphertextBlob,
                           destination_key_id: destinationKeyId
                         })

# Display a readable version of the resulting re-encrypted blob.
puts 'Blob:'
puts resp.ciphertext_blob.unpack('H*')
```

- Pour plus de détails sur l'API, reportez-vous ReEncryptà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn reencrypt_string(
    verbose: bool,
    client: &Client,
    input_file: &str,
    output_file: &str,
    first_key: &str,
    new_key: &str,
) -> Result<(), Error> {
    // Get blob from input file
    // Open input text file and get contents as a string
    // input is a base-64 encoded string, so decode it:
    let data = fs::read_to_string(input_file)
        .map(|input_file| base64::decode(input_file).expect("invalid base 64"))
        .map(Blob::new);

    let resp = client
        .re_encrypt()
        .ciphertext_blob(data.unwrap())
        .source_key_id(first_key)
        .destination_key_id(new_key)
        .send()
        .await?;

    // Did we get an encrypted blob?
    let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);
    let o = &output_file;

    let mut ofile = File::create(o).expect("unable to create file");
    ofile.write_all(s.as_bytes()).expect("unable to write");
```

```
    if verbose {
        println!("Wrote the following to {}:", output_file);
        println!("{}", s);
    } else {
        println!("Wrote base64-encoded output to {}", output_file);
    }

    Ok(())
}
```

- Pour plus de détails sur l'API, voir ReEncryptla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`RetireGrant`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `RetireGrant`.

CLI

AWS CLI

Pour annuler une subvention sur la clé principale d'un client

L'`retire-grant`exemple suivant supprime une autorisation d'une clé KMS.

L'exemple de commande suivant spécifie les `key-id` paramètres `grant-id` et. La valeur du `key-id` paramètre doit être l'ARN de la clé KMS.

```
aws kms retire-grant \
    --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \
    --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette commande ne produit aucun résultat. Pour confirmer que la subvention a été retirée, utilisez la `list-grants` commande.

Pour plus d'informations, consultez la section Retrait et révocation des subventions dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous RetireGrantà la section Référence des AWS CLI commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def retire_grant(self, grant):
        """
        Retires a grant so that it can no longer be used.

        :param grant: The grant to retire.
        """
        try:
            self.kms_client.retire_grant(GrantToken=grant["GrantToken"])
        except ClientError as err:
            logger.error(
```

```
                "Couldn't retire grant %s. Here's why: %s",
                grant["GrantId"],
                err.response["Error"]["Message"],
            )
        else:
            print(f"Grant {grant['GrantId']} retired.")
```

- Pour plus de détails sur l'API, consultez RetireGrantle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`RevokeGrant`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `RevokeGrant`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour révoquer une autorisation sur la clé principale d'un client

L'`revoke-grant`exemple suivant supprime une autorisation d'une clé KMS. L'exemple de commande suivant spécifie les `key-id` paramètres `grant-id` et. La valeur du `key-id` paramètre peut être l'ID de clé ou l'ARN de la clé KMS.

```
aws kms revoke-grant \
    --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette commande ne produit aucun résultat. Pour confirmer que l'autorisation a été révoquée, utilisez la `list-grants` commande.

Pour plus d'informations, consultez la section Retrait et révocation des subventions dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous RevokeGrantà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Revokes a grant for the specified AWS KMS key asynchronously.
    *
    * @param keyId   The ID or key ARN of the AWS KMS key.
    * @param grantId The identifier of the grant to be revoked.
    * @return A {@link CompletableFuture} representing the asynchronous
operation of revoking the grant.
    *         The {@link CompletableFuture} will complete with a {@link
RevokeGrantResponse} object
    *         if the operation is successful, or with a {@code null} value if an
error occurs.
    */
   public CompletableFuture<RevokeGrantResponse> revokeKeyGrantAsync(String
keyId, String grantId) {
       RevokeGrantRequest grantRequest = RevokeGrantRequest.builder()
           .keyId(keyId)
           .grantId(grantId)
           .build();

       CompletableFuture<RevokeGrantResponse> responseFuture =
getAsyncClient().revokeGrant(grantRequest);
       responseFuture.whenComplete((response, exception) -> {
           if (exception == null) {
```

```
                logger.info("Grant ID: [" + grantId + "] was successfully
revoked!");
            } else {
                if (exception instanceof KmsException kmsEx) {
                    if (kmsEx.getMessage().contains("Grant does not exist")) {
                        logger.info("The grant ID '" + grantId + "' does not
exist. Moving on...");
                    } else {
                        throw new RuntimeException("KMS error occurred: " +
kmsEx.getMessage(), kmsEx);
                    }
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
exception.getMessage(), exception);
                }
            }
        });

        return responseFuture;
    }
```

- Pour plus de détails sur l'API, reportez-vous RevokeGrantà la section Référence des AWS SDK for Java 2.x API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $grantId
     * @param string $keyId
     * @return void
     */
```

```
    public function revokeGrant(string $grantId, string $keyId)
    {
        try{
            $this->client->revokeGrant([
                'GrantId' => $grantId,
                'KeyId' => $keyId,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem with revoking the grant: {$caught-
>getAwsErrorMessage()}.\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous RevokeGrantà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
 client.
        """
```

```
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def revoke_grant(self, key_id: str, grant_id: str) -> None:
        """
        Revokes a grant so that it can no longer be used.

        :param key_id: The ARN or ID of the key associated with the grant.
        :param grant_id: The ID of the grant to revoke.
        """
        try:
            self.kms_client.revoke_grant(KeyId=key_id, GrantId=grant_id)
        except ClientError as err:
            logger.error(
                "Couldn't revoke grant %s. Here's why: %s",
                grant_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez RevokeGrantle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ScheduleKeyDeletion`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `ScheduleKeyDeletion`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour planifier la suppression d'une clé KMS gérée par le client.

L'`schedule-key-deletion`exemple suivant planifie la suppression de la clé KMS gérée par le client spécifiée dans les 15 jours.

Le `--key-id` paramètre identifie la clé KMS. Cet exemple utilise une valeur ARN clé, mais vous pouvez utiliser l'ID de clé ou l'ARN de la clé KMS. Le `--pending-window-in-days` paramètre indique la durée de la période d'attente de 7 à 30 jours. Par défaut, le délai d'attente est de 30 jours. Cet exemple indique une valeur de 15, qui indique de AWS supprimer définitivement la clé KMS 15 jours après la fin de la commande.

```
aws kms schedule-key-deletion \
    --key-id arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --pending-window-in-days 15
```

La réponse inclut l'ARN de la clé, l'état de la clé, le délai d'attente (`PendingWindowInDays`) et la date de suppression en heure Unix. Pour afficher la date de suppression en heure locale, utilisez la console AWS KMS. Les clés KMS à l'état `PendingDeletion` clé ne peuvent pas être utilisées dans des opérations cryptographiques.

```
{
    "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "DeletionDate": "2022-06-18T23:43:51.272000+00:00",
    "KeyState": "PendingDeletion",
    "PendingWindowInDays": 15
}
```

Pour plus d'informations, consultez la section Suppression de clés dans le guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous ScheduleKeyDeletionà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Deletes a KMS key asynchronously.
    *
    * <p><strong>Warning:</strong> Deleting a KMS key is a destructive and
potentially dangerous operation.
    * When a KMS key is deleted, all data that was encrypted under the KMS key
becomes unrecoverable.
    * This means that any files, databases, or other data that were encrypted
using the deleted KMS key
    * will become permanently inaccessible. Exercise extreme caution when
deleting KMS keys.</p>
    *
    * @param keyId the ID of the KMS key to delete
    * @return a {@link CompletableFuture} that completes when the key deletion
is scheduled
    */
   public CompletableFuture<Void> deleteKeyAsync(String keyId) {
       ScheduleKeyDeletionRequest deletionRequest =
ScheduleKeyDeletionRequest.builder()
           .keyId(keyId)
           .pendingWindowInDays(7)
           .build();

       return getAsyncClient().scheduleKeyDeletion(deletionRequest)
           .thenRun(() -> {
               logger.info("Key {} will be deleted in 7 days", keyId);
           })
           .exceptionally(throwable -> {
               throw new RuntimeException("Failed to schedule key deletion for
key ID: " + keyId, throwable);
           });
   }
```

- Pour plus de détails sur l'API, reportez-vous ScheduleKeyDeletionà la section Référence des AWS SDK for Java 2.x API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param int $pendingWindowInDays
     * @return void
     */
    public function scheduleKeyDeletion(string $keyId, int $pendingWindowInDays =
 7)
    {
        try {
            $this->client->scheduleKeyDeletion([
                'KeyId' => $keyId,
                'PendingWindowInDays' => $pendingWindowInDays,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem scheduling the key deletion: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous ScheduleKeyDeletionà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
 client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def delete_key(self, key_id: str, window: int) -> None:
        """
        Deletes a list of keys.

        Warning:
        Deleting a KMS key is a destructive and potentially dangerous operation.
 When a KMS key is deleted,
        all data that was encrypted under the KMS key is unrecoverable.

        :param key_id: The ARN or ID of the key to delete.
        :param window: The waiting period, in days, before the KMS key is
 deleted.
        """

        try:
```

```
            self.kms_client.schedule_key_deletion(
                KeyId=key_id, PendingWindowInDays=window
            )
        except ClientError as err:
            logging.error(
                "Couldn't delete key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez ScheduleKeyDeletionle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`Sign`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `Sign`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Exemple 1 : pour générer une signature numérique pour un message

L'`sign`exemple suivant génère une signature cryptographique pour un message court. La sortie de la commande inclut un `Signature` champ codé en base 64 que vous pouvez vérifier à l'aide de la `verify` commande.

Vous devez spécifier un message à signer et un algorithme de signature pris en charge par votre clé KMS asymétrique. Pour obtenir les algorithmes de signature de votre clé KMS, utilisez la `describe-key` commande.

Dans la AWS CLI v2, la valeur du `message` paramètre doit être codée en Base64. Vous pouvez également enregistrer le message dans un fichier et utiliser le `fileb://` préfixe, qui indique à la AWS CLI de lire les données binaires du fichier.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte. L'ID de clé doit représenter une clé KMS asymétrique avec une clé d'utilisation de SIGN_VERIFY.

```
msg=(echo 'Hello World' | base64)

aws kms sign \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://UnsignedMessage \
    --message-type RAW \
    --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256
```

Sortie :

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "Signature": "ABCDEFhpyVYyTxbafE74ccSvEJLJr3zuoV1Hfymz4qv+/
fxmxNLA7SE1SiF8lHw80fKZZ3bJ...",
    "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
}
```

Pour plus d'informations sur l'utilisation de clés KMS asymétriques dans AWS KMS, voir Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

Exemple 2 : pour enregistrer une signature numérique dans un fichier (Linux et macOS)

L'`sign`exemple suivant génère une signature cryptographique pour un message court stocké dans un fichier local. La commande obtient également la `Signature` propriété à partir de la réponse, la décode en Base64 et l'enregistre dans le fichier. ExampleSignature Vous pouvez utiliser le fichier de signature dans une `verify` commande qui vérifie la signature.

La `sign` commande nécessite un message codé en Base64 et un algorithme de signature pris en charge par votre clé KMS asymétrique. Pour obtenir les algorithmes de signature pris en charge par votre clé KMS, utilisez la `describe-key` commande.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte. L'ID de clé doit représenter une clé KMS asymétrique avec une clé d'utilisation de SIGN_VERIFY.

```
echo 'hello world' | base64 > EncodedMessage

aws kms sign \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --message fileb://EncodedMessage \
    --message-type RAW \
    --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
    --output text \
    --query Signature | base64 --decode > ExampleSignature
```

Cette commande ne produit aucun résultat. Cet exemple extrait la `Signature` propriété de la sortie et l'enregistre dans un fichier.

Pour plus d'informations sur l'utilisation de clés KMS asymétriques dans AWS KMS, voir Clés asymétriques dans AWS KMS dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, consultez la section Référence des AWS CLI commandes de connexion.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Asynchronously signs and verifies data using AWS KMS.
```

```
    *
    * <p>The method performs the following steps:
    * <ol>
    *     <li>Creates an AWS KMS key with the specified key spec, key usage, and
origin.</li>
    *     <li>Signs the provided message using the created KMS key and the
RSASSA-PSS-SHA-256 algorithm.</li>
    *     <li>Verifies the signature of the message using the created KMS key
and the RSASSA-PSS-SHA-256 algorithm.</li>
    * </ol>
    *
    * @return a {@link CompletableFuture} that completes with the result of the
signature verification,
    *         {@code true} if the signature is valid, {@code false} otherwise.
    * @throws KmsException if any error occurs during the KMS operations.
    * @throws RuntimeException if an unexpected error occurs.
    */
   public CompletableFuture<Boolean> signVerifyDataAsync() {
       String signMessage = "Here is the message that will be digitally signed";

       // Create an AWS KMS key used to digitally sign data.
       CreateKeyRequest createKeyRequest = CreateKeyRequest.builder()
           .keySpec(KeySpec.RSA_2048)
           .keyUsage(KeyUsageType.SIGN_VERIFY)
           .origin(OriginType.AWS_KMS)
           .build();

       return getAsyncClient().createKey(createKeyRequest)
           .thenCompose(createKeyResponse -> {
               String keyId = createKeyResponse.keyMetadata().keyId();

               SdkBytes messageBytes = SdkBytes.fromString(signMessage,
Charset.defaultCharset());
               SignRequest signRequest = SignRequest.builder()
                   .keyId(keyId)
                   .message(messageBytes)
                   .signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                   .build();

               return getAsyncClient().sign(signRequest)
                   .thenCompose(signResponse -> {
                       byte[] signedBytes =
signResponse.signature().asByteArray();
```

```
                        VerifyRequest verifyRequest = VerifyRequest.builder()
                            .keyId(keyId)

.message(SdkBytes.fromByteArray(signMessage.getBytes(Charset.defaultCharset())))

.signature(SdkBytes.fromByteBuffer(ByteBuffer.wrap(signedBytes)))

.signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                            .build();

                        return getAsyncClient().verify(verifyRequest)
                            .thenApply(verifyResponse -> {
                                return (boolean) verifyResponse.signatureValid();
                            });
                    });
            })
            .exceptionally(throwable -> {
               throw new RuntimeException("Failed to sign or verify data",
throwable);
            });
    }
```

- Pour plus de détails sur l'API, consultez la section Référence de AWS SDK for Java 2.x l'API de connexion.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param string $message
     * @param string $algorithm
```

```
     * @return Result
     */
    public function sign(string $keyId, string $message, string $algorithm)
    {
        try {
            return $this->client->sign([
                'KeyId' => $keyId,
                'Message' => $message,
                'SigningAlgorithm' => $algorithm,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem signing the data: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, consultez la section Référence de AWS SDK pour PHP l'API de connexion.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.
```

```
        :return: An instance of KeyEncrypt initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def sign(self, key_id: str, message: str) -> str:
        """
        Signs a message with a key.

        :param key_id: The ARN or ID of the key to use for signing.
        :param message: The message to sign.
        :return: The signature of the message.
        """
        try:
            return self.kms_client.sign(
                KeyId=key_id,
                Message=message.encode(),
                SigningAlgorithm="RSASSA_PSS_SHA_256",
            )["Signature"]
        except ClientError as err:
            logger.error(
                "Couldn't sign your message. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API Sign in AWS SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`TagResource`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `TagResource`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour ajouter une étiquette à une clé KMS

L'`tag-resource`exemple suivant ajoute `"Purpose":"Test"` des `"Dept":"IT"` balises à une clé KMS gérée par le client. Vous pouvez utiliser de telles balises pour étiqueter les clés KMS et créer des catégories de clés KMS à des fins d'autorisation et d'audit.

Pour spécifier la clé KMS, utilisez le `key-id` paramètre. Cet exemple utilise une valeur d'ID de clé, mais vous pouvez utiliser un ID de clé ou un ARN de clé dans cette commande.

```
aws kms tag-resource \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --tags TagKey='Purpose',TagValue='Test' TagKey='Dept',TagValue='IT'
```

Cette commande ne produit aucun résultat. Pour afficher les balises d'une clé AWS KMS KMS, utilisez la `list-resource-tags` commande.

Pour plus d'informations sur l'utilisation des balises dans AWS KMS, consultez la section Balisage des clés dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous TagResourceà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Asynchronously tags a KMS key with a specific tag.
     *
     * @param keyId the ID of the KMS key to be tagged
     * @return a {@link CompletableFuture} that completes when the tagging
operation is finished
     */
    public CompletableFuture<Void> tagKMSKeyAsync(String keyId) {
        Tag tag = Tag.builder()
            .tagKey("Environment")
            .tagValue("Production")
            .build();

        TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
            .keyId(keyId)
            .tags(tag)
            .build();

        return getAsyncClient().tagResource(tagResourceRequest)
            .thenRun(() -> {
                logger.info("{} key was tagged", keyId);
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to tag the KMS key",
throwable);
            });
    }
```

- Pour plus de détails sur l'API, reportez-vous TagResourceà la section Référence des AWS SDK for Java 2.x API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /***
     * @param string $keyId
     * @param array $tags
     * @return void
     */
    public function tagResource(string $keyId, array $tags)
    {
        try {
            $this->client->tagResource([
                'KeyId' => $keyId,
                'Tags' => $tags,
            ]);
        }catch(KmsException $caught){
            echo "There was a problem applying the tag(s): {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous TagResourceà la section Référence des AWS SDK pour PHP API.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
```

```
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def tag_resource(self, key_id: str, tag_key: str, tag_value: str) -> None:
        """
        Add or edit tags on a customer managed key.

        :param key_id: The ARN or ID of the key to enable rotation for.
        :param tag_key: Key for the tag.
        :param tag_value: Value for the tag.
        """
        try:
            self.kms_client.tag_resource(
                KeyId=key_id, Tags=[{"TagKey": tag_key, "TagValue": tag_value}]
            )
        except ClientError as err:
            logging.error(
                "Couldn't add a tag for the key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez TagResourcele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateAlias`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `UpdateAlias`.

CLI

AWS CLI

Pour associer un alias à une autre clé KMS

L'`update-alias`exemple suivant associe l'alias `alias/test-key` à une autre clé KMS.

Le `--alias-name` paramètre spécifie l'alias. La valeur du nom d'alias doit commencer par `alias/` .Le `--target-key-id` paramètre spécifie la clé KMS à associer à l'alias. Il n'est pas nécessaire de spécifier la clé KMS actuelle pour l'alias.

```
aws kms update-alias \
    --alias-name alias/test-key \
    --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

Cette commande ne produit aucun résultat. Pour trouver l'alias, utilisez la `list-aliases` commande.

Pour plus d'informations, consultez la section Mise à jour des alias dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, reportez-vous UpdateAliasà la section Référence des AWS CLI commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class AliasManager:
    def __init__(self, kms_client):
```

```
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def update_alias(self, alias, current_key_id):
        """
        Updates an alias by assigning it to another key.

        :param alias: The alias to reassign.
        :param current_key_id: The ARN or ID of the key currently associated with
the alias.
        """
        new_key_id = input(
            f"Alias {alias} is currently associated with {current_key_id}. "
            f"Enter another key ID or ARN that you want to associate with
{alias}: "
        )
        if new_key_id != "":
            try:
                self.kms_client.update_alias(AliasName=alias,
TargetKeyId=new_key_id)
            except ClientError as err:
                logger.error(
                    "Couldn't associate alias %s with key %s. Here's why: %s",
                    alias,
                    new_key_id,
                    err.response["Error"]["Message"],
                )
            else:
                print(f"Alias {alias} is now associated with key {new_key_id}.")
        else:
            print("Skipping alias update.")
```

- Pour plus de détails sur l'API, consultez UpdateAliasle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`Verify`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `Verify`.

CLI

AWS CLI

Pour vérifier une signature numérique

La `verify` commande suivante vérifie la signature cryptographique d'un court message codé en Base64. L'identifiant de clé, le message, le type de message et l'algorithme de signature doivent être identiques à ceux utilisés pour signer le message.

Dans la AWS CLI v2, la valeur du `message` paramètre doit être codée en Base64. Vous pouvez également enregistrer le message dans un fichier et utiliser le `fileb://` préfixe, qui indique à la AWS CLI de lire les données binaires du fichier.

La signature que vous spécifiez ne peut pas être codée en base64. Pour obtenir de l'aide pour décoder la signature renvoyée par la `sign` commande, consultez les exemples de `sign` commandes.

La sortie de la commande inclut un `SignatureValid` champ booléen qui indique que la signature a été vérifiée. Si la validation de signature échoue, la `verify` commande échoue également.

Avant d'exécuter cette commande, remplacez l'exemple d'ID de clé par un ID de clé valide provenant de votre AWS compte.

```
aws kms verify \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
```

```
    --message fileb://EncodedMessage \
    --message-type RAW \
    --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
    --signature fileb://ExampleSignature
```

Sortie :

```
{
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "SignatureValid": true,
    "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
}
```

Pour plus d'informations sur l'utilisation de clés KMS asymétriques dans AWS KMS, consultez la section Utilisation de clés asymétriques dans le Guide du développeur du service de gestion des AWS clés.

- Pour plus de détails sur l'API, voir Vérifier dans AWS CLI la référence des commandes.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
 client.
```

```
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)


    def verify(self, key_id: str, message: str, signature: str) -> bool:
        """
        Verifies a signature against a message.

        :param key_id: The ARN or ID of the key used to sign the message.
        :param message: The message to verify.
        :param signature: The signature to verify.
        :return: True when the signature matches the message, otherwise False.
        """
        try:
            response = self.kms_client.verify(
                KeyId=key_id,
                Message=message.encode(),
                Signature=signature,
                SigningAlgorithm="RSASSA_PSS_SHA_256",
            )
            valid = response["SignatureValid"]
            print(f"The signature is {'valid' if valid else 'invalid'}.")
            return valid
        except ClientError as err:
            if err.response["Error"]["Code"] == "SignatureDoesNotMatchException":
                print("The signature is not valid.")
            else:
                logger.error(
                    "Couldn't verify your signature. Here's why: %s",
                    err.response["Error"]["Message"],
                )
                raise
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API Verify in AWS SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Scénarios d' AWS KMS utilisation AWS SDKs

Les exemples de code suivants vous montrent comment implémenter des scénarios courants dans AWS KMS with AWS SDKs. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions AWS KMS ou en les combinant avec d'autres Services AWS. Chaque exemple inclut un lien vers le code source complet, où vous trouverez des instructions sur la configuration et l'exécution du code.

Les scénarios ciblent un niveau d'expérience intermédiaire pour vous aider à comprendre les actions de service dans leur contexte.

Exemples

- Utiliser le chiffrement des tables DynamoDB à l'aide de la version v2 AWS Command Line Interface

Utiliser le chiffrement des tables DynamoDB à l'aide de la version v2 AWS Command Line Interface

L'exemple de code suivant montre comment gérer les options de chiffrement pour les tables DynamoDB.

- Créez une table avec le chiffrement par défaut.
- Créez une table avec une clé CMK gérée par le client.
- Mettez à jour les paramètres de chiffrement des tables.
- Décrivez le chiffrement des tables.

Bash

AWS CLI avec le script Bash

Créez une table avec le chiffrement par défaut.

```
# Create a table with default encryption (AWS owned key)
aws dynamodb create-table \
    --table-name CustomerData \
    --attribute-definitions \
        AttributeName=CustomerID,AttributeType=S \
```

```
    --key-schema \
        AttributeName=CustomerID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --sse-specification Enabled=true,SSEType=KMS
```

Créez une table avec une clé CMK gérée par le client.

```
# Step 1: Create a customer managed key in KMS
aws kms create-key \
    --description "Key for DynamoDB table encryption" \
    --key-usage ENCRYPT_DECRYPT \
    --customer-master-key-spec SYMMETRIC_DEFAULT

# Store the key ID for later use
KEY_ID=$(aws kms list-keys --query "Keys[?contains(KeyArn, 'Key for
 DynamoDB')].KeyId" --output text)

# Step 2: Create a table with the customer managed key
aws dynamodb create-table \
    --table-name SensitiveData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

Mettez à jour le chiffrement des tables.

```
# Update a table to use a different KMS key
aws dynamodb update-table \
    --table-name CustomerData \
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

Décrivez le chiffrement des tables.

```
# Describe the table to see encryption settings
aws dynamodb describe-table \
    --table-name CustomerData \
```

```
    --query "Table.SSEDescription"
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la Référence des commandes AWS CLI .
  - CreateKey
  - CreateTable
  - DescribeTable
  - UpdateTable

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Attestation cryptographique pour AWS Nitro Enclaves

AWS KMS prend en charge l'attestation cryptographique pour AWS Nitro Enclaves. Les applications qui prennent en charge les enclaves AWS Nitro exécutent les opérations AWS KMS cryptographiques suivantes avec un document d'attestation signé pour l'enclave. Ils AWS KMS APIs vérifient que le document d'attestation provient d'une enclave Nitro. Ensuite, au lieu de renvoyer des données en clair dans la réponse, ceux-ci APIs chiffrent le texte en clair avec la clé publique du document d'attestation et renvoient un texte chiffré qui ne peut être déchiffré que par la clé privée correspondante dans l'enclave.

- Decrypt
- DeriveSharedSecret
- GenerateDataKey
- GenerateDataKeyPair
- GenerateRandom

Le tableau suivant montre en quoi la réponse aux demandes d'enclave Nitro diffère de la réponse standard pour chaque opération d'API.

AWS KMS opération	Réponse normale	Réponse pour AWS Nitro Enclaves
`Decrypt`	Renvoie des données en texte brut	Renvoie les données en texte brut par la clé publique à partir du document d'attestation
`DeriveSharedSecret`	Renvoie le secret partagé brut	Renvoie le secret partagé brut chiffré par la clé publique à partir du document d'attesta tion
`GenerateDataKey`	Renvoie une copie en texte brut de la clé de données	Renvoie une copie de la clé de données chiffrées par la clé publique à partir du document d'attestation

AWS KMS opération	Réponse normale	Réponse pour AWS Nitro Enclaves
	(Renvoie également une copie de la clé de données avec une clé KMS)	(Renvoie également une copie de la clé de données avec une clé KMS)
`GenerateDataKeyPair`	Renvoie une copie en texte brut de la clé privée (Renvoie également la clé publique et une copie de la clé privée chiffrée avec une clé KMS)	Renvoie une copie de la clé privée chiffrée par la clé publique à partir du document d'attestation (Renvoie également la clé publique et une copie de la clé privée chiffrée avec une clé KMS)
`GenerateRandom`	Renvoie une chaîne d'octets aléatoire	Renvoie les chaîne d'octets aléatoire chiffrée par la clé publique à partir du document d'attestation

AWS KMS prend en charge les clés de conditions de politique que vous pouvez utiliser pour autoriser ou refuser les opérations d'enclave à l'aide d'une AWS KMS clé basée sur le contenu du document d'attestation. Vous pouvez également suivre les demandes AWS KMS relatives à votre enclave Nitro dans vos AWS CloudTrail journaux.

En savoir plus

- Attestation cryptographique
- AWS KMS clés de condition pour AWS Nitro Enclaves
- Comment appeler AWS KMS APIs une enclave Nitro
- Demandes de surveillance pour les enclaves Nitro

Comment appeler AWS KMS APIs une enclave Nitro

AWS KMS APIs Pour demander une enclave Nitro, utilisez le `Recipient` paramètre de la demande pour fournir le document d'attestation signé pour l'enclave et l'algorithme de chiffrement à utiliser avec la clé publique de l'enclave. Lorsqu'une demande inclut le paramètre `Recipient` avec un document d'attestation signé, la réponse inclut un champ `CiphertextForRecipient` contenant le texte chiffré par la clé publique. Le champ de texte brut est nul ou vide.

Le `Recipient` paramètre doit spécifier un document d'attestation signé provenant d'une enclave AWS Nitro. AWS KMS s'appuie sur la signature numérique du document d'attestation de l'enclave pour prouver que la clé publique contenue dans la demande provient d'une enclave valide. Vous ne pouvez pas fournir votre propre certificat pour signer numériquement le document d'attestation.

Pour spécifier le paramètre `Recipient`, utilisez le SDK d'enclaves Nitro AWS ou n'importe quel SDK AWS . Le SDK AWS Nitro Enclaves, qui n'est pris en charge que dans une enclave Nitro, ajoute automatiquement le `Recipient` paramètre et ses valeurs à chaque demande. AWS KMS Pour demander des enclaves Nitro dans le AWS SDKs, vous devez spécifier le `Recipient` paramètre et ses valeurs. Support pour l'attestation cryptographique de l'enclave Nitro dans le AWS SDKs a été introduit en mars 2023.

AWS KMS prend en charge les clés de conditions de politique que vous pouvez utiliser pour autoriser ou refuser les opérations d'enclave à l'aide d'une AWS KMS clé basée sur le contenu du document d'attestation. Vous pouvez également suivre les demandes AWS KMS relatives à votre enclave Nitro dans vos AWS CloudTrail journaux.

Pour obtenir des informations détaillées sur le `Recipient` paramètre et le champ de `CiphertextForRecipient` réponse AWS, consultez les GenerateRandomrubriques Déchiffrer, DeriveSharedSecret, GenerateDataKeyGenerateDataKeyPair, et dans la référence des AWS Key Management Service API, le SDK AWS Nitro Enclaves ou tout autre SDK. AWS Pour plus d'informations sur la configuration de vos données et de vos clés de données pour le chiffrement, consultez la section Utilisation d'une attestation cryptographique avec AWS KMS.

Demandes de surveillance pour les enclaves Nitro

Vous pouvez utiliser vos AWS CloudTrail journaux pour surveiller le déchiffrement,, DeriveSharedSecretGenerateDataKeyGenerateDataKeyPair, et les GenerateRandomopérations d'une enclave AWS Nitro. Dans ces entrées de journal, le `additionalEventData` champ contient un `recipient` champ contenant l'ID du module (`attestationDocumentModuleId`), le résumé

de l'image (`attestationDocumentEnclaveImageDigest`) et les registres de configuration de la plate-forme (PCRs) provenant du document d'attestation contenu dans la demande. Ces champs ne sont inclus que lorsque le `Recipient` paramètre de la demande spécifie un document d'attestation signé provenant d'une enclave AWS Nitro.

L'ID du module est l'ID d'enclave de l'enclave Nitro. Le condensé d'image est le SHA384 hachage de l'image de l'enclave. Vous pouvez utiliser le résumé de l'image et les valeurs PCR dans les conditions des stratégies de clé et des politiques IAM. Pour plus d'informations à ce sujet PCRs, voir Où obtenir les mesures d'une enclave dans le guide de l'utilisateur de AWS Nitro Enclaves.

Cette section présente un exemple d'entrée de CloudTrail journal pour chacune des demandes d'enclave Nitro prises en charge à AWS KMS.

Decrypt (pour une enclave)

L'exemple suivant montre une entrée de AWS CloudTrail journal d'une opération de déchiffrement pour une enclave AWS Nitro.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-07-27T22:58:24Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "additionalEventData": {
        "recipient": {
```

```
            "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
            "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
            "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
            "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
            "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
            "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
            "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
        }
    },
    "requestID": "b4a65126-30d5-4b28-98b9-9153da559963",
    "eventID": "e5a2f202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

GenerateDataKey (pour une enclave)

L'exemple suivant montre une entrée dans le AWS CloudTrail journal d'une GenerateDataKeyopération pour une enclave AWS Nitro.

```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:52:40Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-1",
```

```
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "numberOfBytes": 32
    },
    "responseElements": null,
    "additionalEventData": {
        "recipient": {
            "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
            "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
            "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
            "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
            "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
            "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
            "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
        }
    },
    "requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

GenerateDataKeyPair (pour une enclave)

L'exemple suivant montre une entrée dans le AWS CloudTrail journal d'une GenerateDataKeyPairopération pour une enclave AWS Nitro.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
            "type": "IAMUser",
            "principalId": "EX_PRINCIPAL_ID",
            "arn": "arn:aws:iam::111122223333:user/Alice",
            "accountId": "111122223333",
            "accessKeyId": "EXAMPLE_KEY_ID",
```

```
            "userName": "Alice"
        },
        "eventTime": "2020-07-27T18:57:57Z",
        "eventSource": "kms.amazonaws.com",
        "eventName": "GenerateDataKeyPair",
        "awsRegion": "us-west-2",
        "sourceIPAddress": "192.0.2.0",
        "userAgent": "AWS Internal",
        "requestParameters": {
            "keyPairSpec": "RSA_3072",
            "encryptionContext": {
                "Project": "Alpha"
            },
            "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        "responseElements": null,
        "additionalEventData": {
            "recipient": {
                "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
                "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
                "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
                "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
                "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
                "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
                "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
            }
        },
        "requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
        "eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
        "readOnly": true,
        "resources": [
            {
                "accountId": "111122223333",
                "type": "AWS::KMS::Key",
                "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        ],
        "eventType": "AwsApiCall",
        "recipientAccountId": "111122223333"
}
```

GenerateRandom (pour une enclave)

L'exemple suivant montre une entrée dans le AWS CloudTrail journal d'une GenerateRandomopération pour une enclave AWS Nitro.


```
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-04T00:52:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateRandom",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData": {
        "recipient": {
            "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
            "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
            "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
            "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
            "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
            "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
            "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
        }
    },
    "requestID": "df1e3de6-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "239cb9f7-ae05-4c94-9221-6ea30eef0442",
    "readOnly": true,
    "resources": [],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Utilisation du AWS KMS chiffrement avec les AWS services

Avec AWS Key Management Service, vous pouvez fournir des clés de chiffrement pour protéger les données dans d'autres AWS services. L'utilisation du AWS KMS chiffrement avec les AWS services fait référence au processus d'intégration AWS KMS à d'autres AWS services pour chiffrer et déchiffrer les données au repos ou en transit. Les développeurs, les administrateurs système et les professionnels de la sécurité peuvent être intéressés par cette rubrique pour sécuriser les données sensibles stockées ou transmises par le biais de AWS services, répondre aux exigences de conformité réglementaire ou mettre en œuvre les meilleures pratiques de chiffrement. Les cas d'utilisation courants incluent le chiffrement de volumes Amazon EBS, de compartiments Amazon S3 et de bases de données Amazon RDS. Les sections suivantes décrivent les étapes de configuration et de gestion des clés de AWS KMS chiffrement pour des AWS services spécifiques, afin de garantir la confidentialité et l'intégrité des données dans votre AWS environnement. Pour la liste complète des AWS services intégrés AWS KMS, voir Intégration des AWS services.

Les rubriques suivantes décrivent en détail l'utilisation de certains services AWS KMS, notamment les clés KMS qu'ils prennent en charge, la manière dont ils gèrent les clés de données, les autorisations dont ils ont besoin et le suivi de l'utilisation des clés KMS par chaque service dans votre compte.

> **Important**
>
> AWS les services intégrés AWS KMS utilisent uniquement des clés KMS de chiffrement symétriques pour chiffrer vos données. Ces services ne prennent pas en charge le chiffrement avec des clés KMS asymétriques. Pour obtenir de l'aide sur la détermination de la symétrie ou de l'asymétrie d'une clé KMS, veuillez consulter Identifier les différents types de clés.

Rubriques

- Comment Amazon Elastic Block Store (Amazon EBS) utilise AWS KMS
- Comment Amazon EMR utilise AWS KMS
- Comment Amazon Redshift utilise AWS KMS

Comment Amazon Elastic Block Store (Amazon EBS) utilise AWS KMS

Cette rubrique décrit en détail comment Amazon Elastic Block Store (Amazon EBS) AWS KMS utilise pour chiffrer les volumes et les instantanés. Pour obtenir des instructions de base sur le chiffrement de volumes Amazon EBS, veuillez consulter Chiffrement Amazon EBS.

Rubriques

Chiffrement Amazon EBS

Lorsque vous attachez un volume Amazon EBS chiffré à un type d'instance Amazon Elastic Compute Cloud (Amazon EC2) compatible, les données stockées au repos sur le volume, les E/S du disque et les instantanés créés à partir du volume sont tous chiffrés. Le chiffrement s'effectue sur les serveurs qui hébergent EC2 les instances Amazon.

Cette fonction est prise en charge sur tous les types de volume Amazon EBS. Vous accédez aux volumes chiffrés de la même manière que vous accédez aux autres volumes ; le chiffrement et le déchiffrement sont gérés de manière transparente et ne nécessitent aucune action supplémentaire de votre part, de votre EC2 instance ou de votre application. Les instantanés de volumes chiffrés sont automatiquement chiffrés et les volumes créés à partir d'instantanés chiffrés sont également automatiquement chiffrés.

Le statut de chiffrement d'un volume EBS est déterminé lorsque vous créez le volume. Vous ne pouvez pas modifier le statut de chiffrement d'un volume existant. Par contre, vous pouvez migrer les données entre des volumes chiffrés et des volumes non chiffrés, et appliquer un nouveau statut de chiffrement lors de la copie d'un instantané.

Amazon EBS prend en charge le chiffrement facultatif par défaut. Vous pouvez activer le chiffrement automatiquement sur tous les nouveaux volumes EBS et les copies instantanées de votre Compte AWS région. Ce paramètre de configuration n'affecte pas les volumes ou instantanés

existants. Pour plus de détails, consultez la section relative au chiffrement Amazon EBS dans le guide de l'utilisateur Amazon EBS.

Utilisation des clés KMS et des clés de données

Lorsque vous créez un volume Amazon EBS., vous spécifiez un AWS KMS key. Par défaut, Amazon EBS utilise la Clé gérée par AWS pour Amazon EBS dans votre compte (`aws/ebs`). Toutefois, vous pouvez spécifier une clé gérée par le client que vous créez et gérez.

Pour utiliser une clé gérée par le client, vous devez autoriser Amazon EBS à utiliser la clé KMS en votre nom. Pour obtenir la liste des autorisations requises, consultez la section Autorisations pour les utilisateurs IAM dans le guide de l' EC2 utilisateur Amazon ou le guide de EC2 l'utilisateur Amazon.

> **Important**
>
> Amazon EBS prend uniquement en charge les clés KMS symétriques. Vous ne pouvez pas utiliser une clé KMS asymétrique pour chiffrer un volume Amazon EBS. Pour obtenir de l'aide sur la détermination de la symétrie ou de l'asymétrie d'une clé KMS, veuillez consulter Identifier les différents types de clés.

Pour chaque volume, Amazon EBS demande de AWS KMS générer une clé de données unique chiffrée sous la clé KMS que vous spécifiez. Amazon EBS stocke la clé de données chiffrée avec le volume. Ensuite, lorsque vous attachez le volume à une EC2 instance Amazon, Amazon EBS appelle AWS KMS pour déchiffrer la clé de données. Amazon EBS utilise la clé de données en texte brut dans la mémoire de l'hyperviseur pour chiffrer toutes les I/O de disque sur le volume. Pour plus de détails, consultez Comment fonctionne le chiffrement EBS dans le guide de l' EC2 utilisateur Amazon ou le guide de EC2 l'utilisateur Amazon.

Contexte du chiffrement Amazon EBS

Dans ses demandes GenerateDataKeyWithoutPlaintextet Decrypt à AWS KMS, Amazon EBS utilise un contexte de chiffrement avec une paire nom-valeur qui identifie le volume ou le snapshot inclus dans la demande. Le nom du contexte de chiffrement ne varie pas.

Un contexte de chiffrement est un ensemble de paires clé-valeur qui contiennent des données non secrètes arbitraires. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, lie AWS KMS cryptographiquement le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez transmettre le même contexte de chiffrement.

Pour tous les volumes et pour les instantanés chiffrés créés avec l'CreateSnapshotopération Amazon EBS, Amazon EBS utilise l'ID du volume comme valeur de contexte de chiffrement. Dans le champ `requestParameters` d'une entrée de journal CloudTrail, le contexte de chiffrement ressemble à ce qui suit :

```
"encryptionContext": {
  "aws:ebs:id": "vol-0cfb133e847d28be9"
}
```

Pour les instantanés chiffrés créés avec l' EC2 CopySnapshotopération Amazon, Amazon EBS utilise l'ID du snapshot comme valeur de contexte de chiffrement. Dans le champ `requestParameters` d'une entrée de journal CloudTrail, le contexte de chiffrement ressemble à ce qui suit :

```
"encryptionContext": {
  "aws:ebs:id": "snap-069a655b568de654f"
}
```

Détection des défaillances Amazon EBS

Pour créer un volume EBS chiffré ou attacher le volume à une EC2 instance, Amazon EBS et l' EC2 infrastructure Amazon doivent être en mesure d'utiliser la clé KMS que vous avez spécifiée pour le chiffrement du volume EBS. Lorsque la clé KMS n'est pas utilisable, par exemple lorsque son état de clé n'est pas `Enabled`, la création ou l'attachement du volume échoue.

Dans ce cas, Amazon EBS envoie un événement à Amazon EventBridge (anciennement CloudWatch Events) pour vous informer de l'échec. Dans EventBridge, vous pouvez établir des règles qui déclenchent des actions automatiques en réponse à ces événements. Pour plus d'informations, consultez Amazon CloudWatch Events pour Amazon EBS dans le guide de l' EC2 utilisateur Amazon, en particulier les sections suivantes :

- Clé de chiffrement non valide pour l'attachement ou le réattachement du volume
- Clé de chiffrement non valide pour la création du volume

Pour résoudre ces problèmes, veillez à ce que la clé KMS spécifiée pour le chiffrement des volumes EBS soit activée. Pour ce faire, consultez d'abord la clé KMS afin de déterminer son état actuel (colonne État dans le AWS Management Console). Ensuite, consultez les informations à l'aide de l'un des liens suivants :

- Si la clé KMS est désactivée, activez-la.
- Si la clé KMS est en attente d'importation, importez-la.
- Si la clé KMS est en attente de suppression, annulez cette suppression.

Utilisation AWS CloudFormation pour créer des volumes Amazon EBS chiffrés

Vous pouvez utiliser AWS CloudFormation pour créer des volumes Amazon EBS chiffrés. Pour plus d'informations, consultez AWS::EC2::Volume dans le Guide de l'utilisateur AWS CloudFormation .

Comment Amazon EMR utilise AWS KMS

Lorsque vous utilisez un cluster Amazon EMR, vous pouvez le configurer pour chiffrer les données au repos avant de les enregistrer dans un emplacement de stockage permanent. Vous pouvez chiffrer des données au repos dans le système de fichiers EMR (EMRFS), sur les volumes de stockage de nœuds de cluster, ou les deux. Pour chiffrer les données au repos, vous pouvez utiliser une AWS KMS key. Les rubriques suivantes expliquent comment un cluster Amazon EMR utilise une clé KMS pour chiffrer des données au repos.

> **Important**
>
> Amazon EMR prend uniquement en charge les clés KMS symétriques. Vous ne pouvez pas utiliser une clé KMS asymétrique pour chiffrer les données au repos dans un cluster Amazon EMR. Pour obtenir de l'aide sur la détermination de la symétrie ou de l'asymétrie d'une clé KMS, veuillez consulter Identifier les différents types de clés.

Les clusters Amazon EMR chiffrent également les données en transit, ce qui signifie que le cluster chiffre les données avant de les envoyer via le réseau. Vous ne pouvez pas utiliser une clé KMS pour chiffrer des données en transit. Pour de plus amples informations, veuillez consulter Chiffrement des données en transit dans le guide de gestion Amazon EMR.

Pour plus d'informations sur toutes les options de chiffrement disponibles dans Amazon EMR, veuillez consulter Options de chiffrement dans le guide de gestion Amazon EMR.

Rubriques

- Chiffrement des données dans le système de fichiers EMR (EMRFS)

- Chiffrement des données sur les volumes de stockage de nœuds de Cluster
- Contexte de chiffrement

Chiffrement des données dans le système de fichiers EMR (EMRFS)

Les clusters Amazon EMR utilisent deux systèmes de fichiers distribués :

- Le système de fichiers distribué Hadoop (HDFS). Le chiffrement HDFS n'utilise pas une clé KMS dans AWS KMS.
- Le système de fichiers EMR (EMRFS). EMRFS est une implémentation de HDFS, qui permet aux clusters Amazon EMR de stocker des données dans Amazon Simple Storage Service (Amazon S3). EMRFS prend en charge quatre options de chiffrement, dont deux utilisent une clé KMS dans AWS KMS. Pour plus d'informations sur les quatre options de chiffrement EMRFS, veuillez consulter Options de chiffrement dans le guide de gestion Amazon EMR.

Les deux options de chiffrement EMRFS qui utilisent une clé KMS font appel aux fonctions de chiffrement suivantes proposées par Amazon S3 :

- Protection des données à l'aide du chiffrement côté serveur avec AWS Key Management Service (SSE-KMS). Le cluster Amazon EMR envoie les données à Simple Storage Service (Amazon S3). Simple Storage Service (Amazon S3) utilise une clé KMS pour chiffrer les données avant de les enregistrer dans un compartiment S3. Pour en savoir plus sur la façon dont cela fonctionne, veuillez consulter Processus de chiffrement des données sur EMRFS avec SSE-KMS.
- Protection des données via le chiffrement côté client (CSE-KMS). Les données d'un Amazon EMR sont chiffrées sous une AWS KMS key avant d'être envoyées à Simple Storage Service (Amazon S3) pour y être stockées. Pour en savoir plus sur la façon dont cela fonctionne, veuillez consulter Processus de chiffrement des données sur EMRFS avec CSE-KMS.

Lorsque vous configurez un cluster Amazon EMR pour chiffrer les données sur EMRFS avec une clé KMS, vous choisissez la clé KMS que vous voulez que le cluster Amazon EMR ou Simple Storage Service (Amazon S3) utilise. Avec SSE-KMS, vous pouvez choisir la Clé gérée par AWS pour Amazon S3 avec l'alias aws/s3, ou une clé symétrique gérée par le client que vous créez. Avec le chiffrement côté client, vous devez choisir une clé symétrique gérée par le client que vous créez. Lorsque vous choisissez une clé gérée par le client, vous devez vous assurer que votre cluster Amazon EMR est autorisé à utiliser la clé KMS. Pour plus d'informations, consultez la section Utilisation à AWS KMS keys des fins de chiffrement dans le guide de gestion Amazon EMR.

Pour le chiffrement côté serveur et côté client, la clé KMS que vous choisissez est la clé racine dans un flux de chiffrement d'enveloppe. Les données sont chiffrées à l'aide d'une clé de données unique cryptée sous la clé KMS dans AWS KMS. Les données chiffrées et une copie chiffrée de leur clé de données sont stockées ensemble en tant qu'objet chiffré unique dans un compartiment S3. Pour plus d'informations sur la façon dont cela fonctionne, consultez les rubriques suivantes.

Rubriques

Processus de chiffrement des données sur EMRFS avec SSE-KMS

Lorsque vous configurez un cluster Amazon EMR pour utiliser SSE-KMS, le processus de chiffrement fonctionne comme suit :

1. Le cluster envoie les données à Amazon S3 pour leur stockage dans un compartiment S3.
2. Amazon S3 envoie une GenerateDataKeydemande à AWS KMS, en spécifiant l'ID de clé KMS que vous avez choisi lorsque vous avez configuré le cluster pour utiliser SSE-KMS. La demande inclut le contexte de chiffrement. Pour plus d'informations, veuillez consulter Contexte de chiffrement.
3. AWS KMS génère une clé de chiffrement de données unique (clé de données), puis envoie deux copies de cette clé de données à Amazon S3. Une copie est non chiffrée (texte brut) et l'autre copie est chiffrée sous la clé KMS.
4. Amazon S3 utilise la clé de données en texte brut pour chiffrer les données reçues à l'étape 1, puis supprime la clé de données en texte brut de la mémoire dès que possible après usage.
5. Amazon S3 stocke ensemble les données chiffrées et la copie chiffrée de la clé de données en tant qu'objet chiffré unique dans un compartiment S3.

Le processus de déchiffrement fonctionne comme suit :

1. Le cluster demande un objet de données chiffré depuis un compartiment S3.
2. Amazon S3 extrait la clé de données chiffrée de l'objet S3, puis envoie la clé de données chiffrée à AWS KMS avec une demande de déchiffrement. La demande comprend un contexte de chiffrement.
3. AWS KMS déchiffre la clé de données chiffrée à l'aide de la même clé KMS que celle utilisée pour la chiffrer, puis envoie la clé de données déchiffrée (texte brut) à Amazon S3.

4. Amazon S3 utilise la clé de données en texte brut pour déchiffrer les données chiffrées, puis supprime la clé de données en texte brut de la mémoire dès que possible après usage.
5. Amazon S3 envoie les données déchiffrées au cluster.

Processus de chiffrement des données sur EMRFS avec CSE-KMS

Lorsque vous configurez un cluster Amazon EMR pour utiliser CSE-KMS, le processus de chiffrement fonctionne comme suit :

1. Lorsqu'il est prêt à stocker des données dans Amazon S3, le cluster envoie une GenerateDataKeydemande à AWS KMS, en spécifiant l'ID de clé KMS que vous avez choisi lorsque vous avez configuré le cluster pour utiliser CSE-KMS. La demande inclut le contexte de chiffrement. Pour plus d'informations, veuillez consulter Contexte de chiffrement.
2. AWS KMS génère une clé de chiffrement de données unique (clé de données), puis envoie deux copies de cette clé de données au cluster. Une copie est non chiffrée (texte brut) et l'autre copie est chiffrée sous la clé KMS.
3. Le cluster utilise la clé de données en texte brut pour chiffrer les données, puis supprime la clé de données en texte brut de la mémoire dès que possible après usage.
4. Le cluster combine les données chiffrées et la copie chiffrée de la clé de données en un objet chiffré unique.
5. Le cluster envoie l'objet chiffré à Amazon S3 pour stockage.

Le processus de déchiffrement fonctionne comme suit :

1. Le cluster demande l'objet de données chiffré depuis un compartiment S3.
2. Amazon S3 envoie l'objet chiffré au cluster.
3. Le cluster extrait la clé de données chiffrée de l'objet chiffré, puis envoie la clé de données chiffrée AWS KMS à une demande de déchiffrement. La demande inclut le contexte de chiffrement.
4. AWS KMS déchiffre la clé de données chiffrée à l'aide de la même clé KMS que celle utilisée pour la chiffrer, puis envoie la clé de données déchiffrée (texte brut) au cluster.
5. Le cluster utilise la clé de données en texte brut pour déchiffrer les données chiffrées, puis supprime la clé de données en texte brut de la mémoire dès que possible après usage.

Chiffrement des données sur les volumes de stockage de nœuds de Cluster

Un cluster Amazon EMR est un ensemble d'instances Amazon Elastic Compute Cloud EC2 (Amazon). Chaque instance du cluster est appelée nœud de cluster ou nœud. Chaque nœud peut avoir deux types de volumes de stockage : des volumes de stockage d'instance et des volumes Amazon Elastic Block Store (Amazon EBS). Vous pouvez configurer le cluster pour utiliser Linux Unified Key Setup (LUKS) pour chiffrer les deux types de volumes de stockage sur les nœuds (mais pas le volume de démarrage de chaque nœud). Il s'agit du chiffrement de disque local.

Lorsque vous activez le chiffrement de disque local pour un cluster, vous pouvez choisir de chiffrer la clé LUKS avec une clé KMS dans AWS KMS. Vous devez choisir une clé gérée par le client que vous créez ; vous ne pouvez pas utiliser une Clé gérée par AWS. Si vous choisissez une clé gérée par le client, vous devez vous assurer que votre cluster Amazon EMR est autorisé à utiliser la clé KMS. Pour plus d'informations, consultez la section Utilisation à AWS KMS keys des fins de chiffrement dans le guide de gestion Amazon EMR.

Lorsque vous activez le chiffrement de disque local à l'aide d'une clé KMS, le processus de chiffrement fonctionne comme ceci :

1. Lorsque chaque nœud de cluster est lancé, il envoie une GenerateDataKeydemande à AWS KMS, spécifiant l'ID de clé KMS que vous avez choisi lorsque vous avez activé le chiffrement de disque local pour le cluster.
2. AWS KMS génère une clé de chiffrement de données unique (clé de données), puis envoie deux copies de cette clé de données au nœud. Une copie est non chiffrée (texte brut) et l'autre copie est chiffrée sous la clé KMS.
3. Le nœud utilise une version encodée en base 64 de la clé de données en texte brut comme mot de passe qui protège la clé LUKS. Le nœud enregistre la copie chiffrée de la clé de données sur le volume de démarrage.
4. Si le nœud redémarre, le nœud redémarré envoie la clé de données chiffrée à AWS KMS avec une demande de déchiffrement.
5. AWS KMS déchiffre la clé de données chiffrée à l'aide de la même clé KMS que celle utilisée pour la chiffrer, puis envoie la clé de données déchiffrée (texte brut) au nœud.
6. Le nœud utilise la version encodée en base 64 de la clé de données en texte brut comme mot de passe pour déverrouiller la clé LUKS.

Contexte de chiffrement

Chaque AWS service intégré AWS KMS peut spécifier un contexte de chiffrement lorsque le service est utilisé AWS KMS pour générer des clés de données ou pour chiffrer ou déchiffrer des données. Le contexte de chiffrement est une information authentifiée supplémentaire AWS KMS utilisée pour vérifier l'intégrité des données. Quand service spécifie un contexte de chiffrement pour une opération de chiffrement, il doit spécifier le même contexte de chiffrement pour l'opération de déchiffrement correspondante, sinon le déchiffrement échouera. Le contexte de chiffrement est également écrit dans les fichiers AWS CloudTrail journaux, ce qui peut vous aider à comprendre pourquoi une clé KMS spécifique a été utilisée.

La section suivante décrit le contexte de chiffrement utilisé dans chaque scénario de chiffrement Amazon EMR qui utilise une clé KMS.

Contexte de chiffrement pour le chiffrement EMRFS avec SSE-KMS

Avec SSE-KMS, le cluster Amazon EMR envoie les données à Amazon S3, puis Amazon S3 utilise une clé KMS pour chiffrer les données avant de les enregistrer dans un compartiment S3. Dans ce cas, Amazon S3 utilise le nom de ressource Amazon (ARN) de l'objet S3 comme contexte de chiffrement pour chaque demande GenerateDataKeyde déchiffrement à AWS KMS laquelle il envoie. L'exemple suivant montre une représentation JSON du contexte de chiffrement qu'Amazon S3 utilise.

```
{ "aws:s3:arn" : "arn:aws:s3:::S3_bucket_name/S3_object_key" }
```

Contexte de chiffrement pour le chiffrement EMRFS avec CSE-KMS

Avec CSE-KMS, le cluster Amazon EMR utilise une clé KMS pour chiffrer les données avant de les envoyer à Amazon S3 pour stockage. Dans ce cas, le cluster utilise l'Amazon Resource Name (ARN) de la clé KMS comme contexte de chiffrement pour chaque GenerateDataKeydemande de déchiffrement à AWS KMS laquelle il envoie. L'exemple suivant montre une représentation JSON du contexte de chiffrement que le cluster utilise.

```
{ "kms_cmk_id" : "arn:aws:kms:us-
east-2:111122223333:key/0987ab65-43cd-21ef-09ab-87654321cdef" }
```

Contexte de chiffrement pour le chiffrement de disque Local avec LUKS

Lorsqu'un cluster Amazon EMR utilise le chiffrement de disque local avec LUKS, les nœuds du cluster ne spécifient pas le contexte de chiffrement des demandes GenerateDataKeyet de déchiffrement auxquelles ils envoient. AWS KMS

Comment Amazon Redshift utilise AWS KMS

Cette rubrique explique comment Amazon Redshift crypte AWS KMS les données.

Rubriques

- Chiffrement Amazon Redshift
- Contexte de chiffrement

Chiffrement Amazon Redshift

Un entrepôt des données Amazon Redshift est un ensemble de ressources informatiques appelées nœuds, qui sont organisées en un groupe appelé cluster. Chaque cluster exécute un moteur Amazon Redshift et contient une ou plusieurs bases de données.

Amazon Redshift utilise une architecture à quatre niveaux de clés pour le chiffrement. Cette architecture se compose de clés de chiffrement des données, d'une clé de base de données, d'une clé de cluster et d'une clé racine. Vous pouvez utiliser un AWS KMS key comme clé racine.

Les clés de chiffrement de données chiffrent les blocs de données contenus dans le cluster. Chaque bloc de données se voit attribuer une clé AES-256 générée de façon aléatoire. Ces clés sont chiffrées à l'aide de la clé de base de données du cluster.

La clé de base de données chiffre les clés de chiffrement de données contenues dans le cluster. La clé de base de données est une clé AES-256 générée de façon aléatoire. Elle est stockée sur disque dans un autre réseau que celui du cluster Amazon Redshift et transmise au cluster via un canal sécurisé.

La clé de cluster chiffre la clé de base de données du cluster Amazon Redshift. Vous pouvez utiliser AWS KMS AWS CloudHSM, ou un module de sécurité matérielle (HSM) externe pour gérer la clé de cluster. Consultez la documentation relative au chiffrement de base de données Amazon Redshift pour plus d'informations.

Vous pouvez demander le chiffrement en cochant la case appropriée dans la console Amazon Redshift. Vous pouvez spécifier une clé gérée par le client en la choisissant dans la liste qui s'affiche sous la zone de chiffrement. Si vous ne spécifiez pas de clé gérée par le client, Amazon Redshift utilise la Clé gérée par AWS pour Amazon Redshift sous votre compte.

> **Important**
>
> Amazon Redshift prend uniquement en charge les clés KMS de chiffrement symétriques. Vous ne pouvez pas utiliser une clé KMS asymétrique dans un flux de travail de chiffrement Amazon Redshift. Pour obtenir de l'aide sur la détermination de la symétrie ou de l'asymétrie d'une clé KMS, consultez Identifier les différents types de clés.

Contexte de chiffrement

Chaque service intégré AWS KMS spécifie un contexte de chiffrement lors de la demande de clés de données, du chiffrement et du déchiffrement. Le contexte de chiffrement est constitué de données authentifiées supplémentaires (AAD) AWS KMS utilisées pour vérifier l'intégrité des données. Autrement dit, lorsqu'un contexte de chiffrement est spécifié pour une opération de chiffrement, le service le spécifie également pour l'opération de déchiffrement, ou le déchiffrement échoue. Amazon Redshift utilise l'ID de cluster et le temps de création du contexte de chiffrement. Dans le `requestParameters` champ d'un fichier CloudTrail journal, le contexte de chiffrement sera similaire à celui-ci.

```
"encryptionContext": {
    "aws:redshift:arn": "arn:aws:redshift:region:account_ID:cluster:cluster_name",
    "aws:redshift:createtime": "20150206T1832Z"
},
```

Vous pouvez effectuer une recherche sur le nom du cluster dans vos CloudTrail journaux pour comprendre quelles opérations ont été effectuées à l'aide d'une AWS KMS key (clé KMS). Les opérations incluent le chiffrement du cluster, le déchiffrement du cluster et la génération de clés de données.

AWS KMS Référence

Le matériel de référence suivant fournit des informations utiles sur l'utilisation et la gestion des clés KMS.

- Référence de type de clé. Répertorie le type de clé KMS qui prend en charge chaque opération d'API AWS KMS

  Pour trouver : Puis-je activer et désactiver une clé KMS de signature RSA ?

- Tableau d'états de clé Indique comment l'état de clé d'une clé KMS affecte son utilisation dans les opérations d'API AWS KMS

  Pour trouver : Puis-je modifier l'alias d'une clé KMS en attente de suppression ?

- AWS KMSRéférence des autorisations d'API. Fournit des informations sur les autorisations requises pour chaque opération AWS KMS d'API.

  À rechercher : Puis-je utiliser GetKeyPolicyune clé d'un autre AWS compte ? Puis-je octroyer l'autorisation `kms:Decrypt` dans une politique IAM ?

  - ViaService référence. Répertorie les services AWS prenant en charge la clé de condition `kms:ViaService`.

    À rechercher : puis-je utiliser la clé de `kms:ViaService` condition pour autoriser une autorisation uniquement lorsqu'elle provient d'Amazon ElastiCache ? Qu'en est-il d'Amazon Neptune ?

- AWS KMS tarification. Répertorie et explique le prix des clés KMS.

  Pour trouver : Combien coûte l'utilisation de mes clés asymétriques ?

- AWS KMS demander des quotas. Répertorie les quotas par seconde pour les demandes d' AWS KMS API dans chaque compte et région.

  Pour trouver : Combien de demandes Decrypt puis-je exécuter chaque seconde ? Combien de demandes Decrypt puis-je exécuter sur des clés KMS dans mon magasin de clés personnalisé ?

- AWS KMS quotas de ressources. Répertorie les quotas sur les ressources AWS KMS .

  Pour trouver : De combien de clés KMS puis-je disposer dans chaque région de mon compte ? De combien d'alias puis-je disposer sur chaque clé KMS ?

- AWS services intégrés à AWS KMS. Répertorie les AWS services qui utilisent des clés KMS pour protéger les ressources qu'ils créent, stockent et gèrent.

  Pour trouver : Amazon Connect utilise-t-il des clés KMS pour protéger mes ressources Connect ?

États clés des AWS KMS clés

Un a AWS KMS key toujours un état clé. Les opérations sur la clé KMS et son environnement peuvent modifier l'état de la clé. L'état clé peut changer soit de façon transitoire, soit jusqu'à ce qu'une autre opération change son état clé. Ces opérations sont effectuées soit de manière asynchrone, soit par un appel d'API.

Le tableau de cette section montre comment les états clés affectent les appels aux opérations AWS KMS d'API. En raison de son état clé, une opération sur une clé KMS devrait réussir (#), échouer (X), ou ne réussir que dans certaines conditions (?). Le résultat est souvent différent pour les clés KMS avec des éléments de clé importés.

Ce tableau inclut uniquement les opérations d'API qui utilisent une clé KMS existante. Les autres opérations, telles que CreateKeyet ListKeys, sont omises.

Rubriques

- États de clé et types de clés KMS
- Tableau d'état de clé

États de clé et types de clés KMS

Le type de la clé KMS détermine les états de clé qu'elle peut avoir.

- Toutes les clés KMS peuvent être à l'état `Enabled`, `Disabled` et `PendingDeletion`.
- La plupart des clés KMS sont créées dans l'état `Enabled`. Les clés avec des éléments de clé importés sont créées dans l'état `PendingImport`.
- L'état `PendingImport` s'applique uniquement aux clés KMS avec des éléments de clé importés. Lorsqu'un élément clé d'une clé importée est supprimé ou qu'il expire, l'état passe de `Enabled` à`PendingImport`.
- L'état `Unavailable` s'applique uniquement à une clé KMS dans un magasin de clés personnalisé. Une clé KMS dans un magasin de AWS CloudHSM clés se produit `Unavailable` lorsque le

magasin de clés personnalisé est intentionnellement déconnecté de son AWS CloudHSM cluster. Une clé KMS dans un magasin de clés externe est `Unavailable` lorsque le magasin de clés personnalisé est intentionnellement déconnecté de son proxy de magasin de clés externe. Vous pouvez afficher et gérer les clés KMS indisponibles, mais vous ne pouvez pas les utiliser dans les opérations de chiffrement.

L'état d'une clé KMS dans un magasin de clés personnalisé n'est pas affecté par les modifications apportées à sa clé de sauvegarde. Une clé KMS dans un magasin de AWS CloudHSM clés n'est pas affectée par les modifications apportées à son contenu clé associé dans le AWS CloudHSM cluster. Une clé KMS dans un magasin de clés externe n'est pas affectée par les modifications apportées à sa clé externe dans un gestionnaire de clés externe. Si la clé de sauvegarde est désactivée ou supprimée, l'état de la clé KMS ne change pas, mais les opérations cryptographiques utilisant la clé KMS échouent.

- Les états de clé `Creating`, `Updating` et `PendingReplicaDeletion` s'appliquent uniquement aux clés multi-région.
  - Une clé de réplica multi-région se trouve dans l'état de clé `Creating` durant sa création. Ce processus est peut-être toujours en cours une fois l'ReplicateKeyopération terminée. Lorsque le processus de réplication est terminé, la clé de réplica se trouve dans l'état `Enabled` ou `PendingImport`.
  - Les clés multi-région se trouvent à l'état transitoire `Updating` lorsque la région principale est en cours de mise à jour. Ce processus est peut-être toujours en cours une fois l'UpdatePrimaryRegionopération terminée. Une fois le processus de mise à jour terminé, les clés principales et de réplica reprennent l'état de clé `Enabled`.
  - Lorsque vous planifiez la suppression d'une clé principale multi-région dotée de clés de réplica, la clé principale se trouve à l'état `PendingReplicaDeletion` jusqu'à ce que toutes ses clés de réplica soient supprimées. Puis, son état passe à `PendingDeletion`. Pour plus de détails, veuillez consulter Deleting multi-Region keys.

Tableau d'état de clé

Le tableau suivant montre comment l'état de clé d'une clé KMS affecte les opérations AWS KMS .

Les descriptions des notes de bas de page numérotées ([n]) sont à la fin de cette rubrique.

> **Note**
>
> Vous devrez peut-être faire défiler horizontalement ou verticalement pour voir toutes les données de ce tableau.

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
CancelKeyDeletion	 [4]	 [4]		 [4]	 [4], [13]	 [4]	 [4]
CreateAlias			 [3]				
CreateGrant		 [1]	 [2] ou [3]	 [5]		 [14]	
Decrypt		 [1]	 [2] ou [3]	 [5]	 [11]	 [14]	
DeleteAlias							

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
DeleteImportedKeyMaterial	✓ [9]	✓ [9]	✓ [9]	✓	N/A	⊗ [14]	⊗ [15]
DeriveSharedSecret	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	N/A	⊗ [14]	✓
DescribeKey	✓	✓	✓	✓	✓	✓	✓
DisableKey	✓	✓	⊗ [3]	⊗ [5]	✓ [12]	⊗ [14]	⊗ [15]
DisableKeyRotation	? [7]	⊗ [1] ou [7]	⊗ [3] ou [7]	⊗ [6]	⊗ [7]	⊗ [14]	? [7]
EnableKey	✓	✓	⊗ [3]	⊗ [5]	✓ [12]	⊗ [14]	⊗ [15]

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
EnableKeyRotation	? [7]	⊗ [1] ou [7]	⊗ [3] ou [7]	⊗ [6]	⊗ [7]	⊗ [14]	? [7]
Encrypt	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	⊗ [11]	⊗ [14]	✓
GenerateDataKey	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	⊗ [11]	⊗ [14]	✓
GenerateDataKeyPair	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	⊗ [7]	⊗ [14]	✓
GenerateDataKeyPairWithoutPlaintext	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	⊗ [7]	⊗ [14]	✓
GenerateDataKeyWithoutPlaintext	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	⊗ [11]	⊗ [14]	✓

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
GenerateMac	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	N/A	⊗ [14]	✓
GetKeyPolicy	✓	✓	✓	✓	✓	✓	✓
GetKeyRotationStatus	? [7]	? [7]	? [7]	⊗ [6]	⊗ [7]	? [7]	? [7]
GetParametersForImport	? [9]	? [9]	⊗ [8] ou [9]	✓	⊗ [9]	⊗ [14]	⊗ [15]
GetPublicKey	✓	✓	⊗ [2] ou [3]	✓	N/A	⊗ [14]	✓
ImportKeyMaterial	? [9]	? [9]	⊗ [9]	✓	⊗ [9]	⊗ [14]	✓
ListAliases	✓	✓	✓	✓	✓	✓	✓

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
ListGrants	✓	✓	✓	✓	✓	✓	✓
ListKeyPolicies	✓	✓	✓	✓	✓	✓	✓
ListKeyRotations	? [7]	? [7]	? [7]	⊗ [6]	⊗ [7]	? [7]	? [7]
ListResourceTags	✓	✓	✓	✓	✓	✓	✓
PutKeyPolicy	✓	✓	✓	✓	✓	✓	✓
ReEncrypt	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	⊗ [11]	⊗ [14]	✓
ReplicateKey	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	N/A	⊗ [14]	⊗ [15]
RetireGrant	✓	✓	✓	✓	✓	✓	✓

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
RevokeGrant	✓	✓	✓	✓	✓	✓	✓
RotateKeyOnDemand	? [7]	⊗ [1] ou [7]	⊗ [3] ou [7]	⊗ [5]	⊗ [7]	⊗ [14]	? [7]
ScheduleKeyDeletion	✓	✓	⊗ [3]	✓	✓	✓	⊗ [15]
Sign (Signer)	✓	⊗ [1]	⊗ [2] ou [3]	⊗ [5]	N/A	⊗ [14]	✓
TagResource	✓	✓	⊗ [3]	✓	✓	✓	✓
UntagResource	✓	✓	⊗ [3]	✓	✓	✓	✓

« Hello, World! »	Activé	Désactivées	Suppression en attente Suppression du réplica en attente	Importation en attente	Unavailable	Création	Mise à jour
UpdateAlias	✓	✓	? [10]	✓	✓	✓	✓
UpdateKeyDescription	✓	✓	⊘ [3]	✓	✓	✓	✓
UpdatePrimaryRegion	✓	⊘ [1]	⊘ [2] ou [3]	⊘ [5]	N/A	⊘ [14]	✓
Vérification	✓	⊘ [1]	⊘ [2] ou [3]	⊘ [5]	N/A	⊘ [14]	✓
VerifyMac	✓	⊘ [1]	⊘ [2] ou [3]	⊘ [5]	N/A	⊘ [14]	✓

Détails de la table

- [1] `DisabledException: <key ARN> is disabled.`
- [2] `DisabledException: <key ARN> is pending deletion (or pending replica deletion).`

- [3] `KMSInvalidStateException: <key ARN> is pending deletion (or pending replica deletion).`
- [4] `KMSInvalidStateException: <key ARN> is not pending deletion (or pending replica deletion).`
- [5] `KMSInvalidStateException: <key ARN> is pending import because no key material has ever been imported or one of the imported key materials is deleted or expired.`
- [6] `UnsupportedOperationException: <key ARN> origin is EXTERNAL which is not valid for this operation.`
- [7] Si la clé KMS se trouve dans un magasin de clés personnalisé :`UnsupportedOperationException`.
- [8] Si la clé KMS comporte des éléments de clé importés : `KMSInvalidStateException`
- [9] Si la clé KMS ne peut pas contenir de matériel clé importé :`UnsupportedOperationException`.
- [10] Si la clé KMS source est en attente de suppression, la commande réussit. Si la clé KMS de destination est en attente de suppression, la commande échoue avec l'erreur suivante : `KMSInvalidStateException : <key ARN> is pending deletion.`
- [11] `KMSInvalidStateException: <key ARN> is unavailable.` Vous ne pouvez pas effectuer cette opération sur une clé KMS indisponible.
- [12] L'opération aboutit, mais l'état de la clé KMS ne change pas jusqu'à ce qu'elle devienne disponible.
- [13] Même si une clé KMS d'un magasin de clés personnalisé est en attente de suppression, son état de clé demeure `PendingDeletion`, même si la clé KMS devient indisponible. Cela vous permet d'annuler la suppression de la clé KMS à tout moment au cours de la période d'attente.
- [14] `KMSInvalidStateException: <key ARN> is creating.` AWS KMS lance cette exception lors de la réplication d'une clé multirégionale ()`ReplicateKey`.
- [15] `KMSInvalidStateException: <key ARN> is updating.` AWS KMS lance cette exception lors de la mise à jour de la région principale d'une clé multirégionale (`UpdatePrimaryRegion`).

Référence des types de clés

AWS KMS prend en charge différentes fonctionnalités pour différents types de clés KMS. Par exemple, vous ne pouvez utiliser que des clés KMS de chiffrement symétriques pour générer des

clés de données symétriques et des paires de clés de données asymétriques. En outre, l'importation d'un élément de clé et la rotation automatique des clés sont prises en charge uniquement pour les clés KMS de chiffrement symétriques. Vous pouvez créer uniquement des clés KMS de chiffrement symétriques dans un magasin de clés personnalisé.

Cette référence comprend deux tableaux.

- Le tableau des types de clés répertorie les AWS KMS opérations valides pour les clés KMS de chiffrement symétriques, les clés KMS asymétriques et les clés KMS HMAC.
- Le tableau de fonctionnalités spéciales répertorie les opérations AWS KMS qui sont valides pour les clés KMS multi-régions, les clés KMS avec des éléments de clé importés et les clés KMS des magasins de clés personnalisés.

Tableau des types de clé

Il peut être nécessaire de faire défiler horizontalement ou verticalement pour afficher toutes les données de ce tableau.

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
CancelKeyDeletion	Oui	Oui	Oui	Oui	Oui
CreateAlias	Oui	Oui	Oui	Oui	Oui
CreateGrant	Oui	Oui	Oui	Oui	Oui
CreateKey	Oui	Oui	Oui	Oui	Oui
Decrypt	Oui	Non	Oui	Non	Non
DeleteAlias	Oui	Oui	Oui	Oui	Oui
DeleteImportedKeyMaterial	Oui	Oui	Oui	Oui	Oui

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
Valable uniquement sur les clés KMS avec des éléments de clé importés (`Origin` est `EXTERNAL`).					
DeriveSharedSecret	Non	Non	Non	Non	Oui
DescribeKey	Oui	Oui	Oui	Oui	Oui
DisableKey	Oui	Oui	Oui	Oui	Oui
DisableKeyRotation	Oui Valable uniquement sur les clés KMS contenant le (s) matériel (`Origin`s AWS KMS clé (`sAWS_KM`	Non	Non	Non	Non
EnableKey	Oui	Oui	Oui	Oui	Oui

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYP DECRYP	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
EnableKeyRotation	Oui Valable uniquement sur les clés KMS contenant le (s) matériel (`Origin`s AWS KMS clé (`sAWS_KM`	Non	Non	Non	Non
Encrypt	Oui	Non	Oui	Non	Non
GenerateDataKey	Oui	Non	Non	Non	Non
GenerateDataKeyPair Génère une paire de clés de données asymétriques qui est protégée par une clé KMS de chiffrement symétrique.	Oui Non valable sur les clés KMS des magasins de clés personnalisés.	Non	Non	Non	Non

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
GenerateDataKeyPairWithoutPlaintext Génère une paire de clés de données asymétriques qui est protégée par une clé KMS de chiffrement symétrique.	Oui Non valable sur les clés KMS des magasins de clés personnalisés.	Non	Non	Non	Non
GenerateDataKeyWithoutPlaintext	Oui	Non	Non	Non	Non
GenerateMac	Non	Oui	Non	Non	Non
GetKeyPolicy	Oui	Oui	Oui	Oui	Oui
GetKeyRotationStatus	Oui	Oui (KeyRotationEnabled sera toujours `false`.)	Oui (KeyRotationEnabled sera toujours `false`.)	Oui (KeyRotationEnabled sera toujours `false`.)	Oui (KeyRotationEnabled sera toujours `false`.)

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
GetParametersForImport Valable uniquement sur les clés KMS avec des éléments de clé importés (`Origin` est `EXTERNAL`).	Oui	Oui	Oui	Oui	Oui
GetPublicKey	Non	Non	Oui	Oui	Oui
ImportKeyMaterial Valable uniquement sur les clés KMS avec des éléments de clé importés (`Origin` est `EXTERNAL`).	Oui	Oui	Oui	Oui	Oui
ListAliases	Oui	Oui	Oui	Oui	Oui
ListGrants	Oui	Oui	Oui	Oui	Oui
ListKeyPolicies	Oui	Oui	Oui	Oui	Oui

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
ListKeyRotations	Oui	Oui (Le `Rotation` champ sera toujours vide ou nul.)	Oui (Le `Rotation` champ sera toujours vide ou nul.)	Oui (Le `Rotations` champ sera toujours vide ou nul.)	Oui (Le `Rotations` champ sera toujours vide ou nul.)
ListResourceTags	Oui	Oui	Oui	Oui	Oui
ListRetirableGrants	Oui	Oui	Oui	Oui	Oui
PutKeyPolicy	Oui	Oui	Oui	Oui	Oui
ReEncrypt	Oui	Non	Oui	Non	Non
ReplicateKey - Valide uniquement sur les clés multi-région	Oui	Oui	Oui	Oui	Oui
RetireGrant	Oui	Oui	Oui	Oui	Oui
RevokeGrant	Oui	Oui	Oui	Oui	Oui

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
RotateKeyOnDemand	Oui Valable uniquement sur les clés KMS de chiffrement symétrique gérées par le client avec `AWS_KMS` ou origine. `EXTERNAL`	Non	Non	Non	Non
ScheduleKeyDeletion	Oui	Oui	Oui	Oui	Oui
Sign (Signer)	Non	Non	Non	Oui	Non
TagResource	Oui	Oui	Oui	Oui	Oui
UntagResource	Oui	Oui	Oui	Oui	Oui

AWS KMS Fonctionnement de l'API	Clés KMS de chiffrement symétrique	Clés KMS HMAC	Clés KMS asymétriques (ENCRYPT DECRYPT	Clés KMS asymétriques (SIGN_VERIFY)	Clés KMS asymétriques (KEY_AGREEMENT)
UpdateAlias La clé KMS actuelle et la nouvelle clé KMS doivent être du même type (toutes deux soit symétriques, soit asymétriques, soit HMAC) et avoir la même utilisation de clé.	Oui	Oui	Oui	Oui	Oui
UpdateKeyDescription	Oui	Oui	Oui	Oui	Oui
UpdateReplicaRegion - Valide uniquement sur les clés multi-région	Oui	Oui	Oui	Oui	Oui
Vérification	Non	Non	Non	Oui	Non
VerifyMac	Non	Oui	Non	Non	Non

Tableau des fonctionnalités spéciales

Ce tableau indique les opérations AWS KMS d'API prises en charge sur chaque type de clé à usage spécifique.

En lisant ce tableau, soyez attentif aux interactions suivantes :

- Clés multi-région:
  - Les clés multi-régions peuvent être des clés KMS de chiffrement symétriques, des clés KMS asymétriques, des clés KMS HMAC et des clés KMS avec éléments de clé importés.
  - Vous ne pouvez pas créer de clés multi-région dans un magasin de clés personnalisé.
- Éléments de clé importés
  - Vous pouvez importer des éléments de clé pour des clés KMS de chiffrement symétriques, des clés KMS asymétriques et des clés KMS HMAC.
  - Vous pouvez créer des clés multi-région avec des éléments de clé importés.
  - Vous ne pouvez pas créer de clés avec des éléments de clé importés dans un magasin de clés personnalisé.
  - La rotation automatique des clés (`EnableKeyRotation`, `DisableKeyRotation`) n'est pas prise en charge pour les clés KMS avec des éléments de clé importés.
  - La rotation des clés à la demande (`RotateKeyOnDemand`) est prise en charge pour les clés KMS de chiffrement symétrique à région unique avec du matériel clé importé.
- Magasins de clés personnalisés
  - Les magasins de clés personnalisés ne prennent en charge que les clés KMS de chiffrement symétriques.
  - Les opérations symétriques sur des paires de clés asymétriques (`GenerateDataKeyPair`, `GenerateDataKeyPairWithoutPlaintext`) ne sont pas prises en charge sur les clés KMS dans les magasins de clés personnalisés.
  - La rotation automatique des clés (`EnableKeyRotation`, `DisableKeyRotation`) n'est pas prise en charge sur les clés KMS dans les magasins de clés personnalisés.
  - Vous ne pouvez pas créer de clés multi-région dans les magasins de clés personnalisés.

Il peut être nécessaire de faire défiler horizontalement ou verticalement pour afficher toutes les données de ce tableau.

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
CancelKeyDeletion	✓	✓	✓
CreateAlias	✓	✓	✓
CreateGrant	✓	✓	✓
CreateKey Vous pouvez utiliser `CreateKey` pour créer une clé primaire multi-régions, une clé KMS avec des éléments de clé importés ou une clé KMS dans un magasin de clés personnalisé. Pour créer une clé de réplica multi-régions, utilisez `ReplicateKey` .	✓	✓	✓
Decrypt	✓ Valable uniquement lorsque `KeyUsage` est `ENCRYPT_DECRYPT`	✓	✓
DeleteAlias	✓	✓	✓
DeleteImportedKeyMaterial	✓	✓	⊗

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
	Valable uniquement pour les clés avec éléments de clé importés (`Origin` est `EXTERNAL`)		
DeriveSharedSecret	 Valable uniquement (quand `KeyUsage` c'est le cas `KEY_AGREEMENT`)	 Valable uniquement (quand `KeyUsage` c'est le cas `KEY_AGREEMENT`)	
DescribeKey			
DisableKey			

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
DisableKeyRotation	✓ Valable uniquement sur les clés de chiffrement symétriques dont le contenu AWS KMS clé `Origin` est (est`AWS_KMS`).	⊗	⊗
EnableKey	✓ Valable uniquement sur les clés KMS de chiffrement symétrique	✓	✓
EnableKeyRotation	✓ Valable uniquement sur les clés de chiffrement symétriques dont le contenu AWS KMS clé `Origin` est (est`AWS_KMS`).	⊗	⊗

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
Encrypt	✓ Valable uniquement lorsque `KeyUsage` est `ENCRYPT_DECRYPT`	✓	✓
GenerateDataKey	✓ Valable uniquement sur les clés KMS de chiffrement symétrique	✓	✓
GenerateDataKeyPair	✓ Valable uniquement sur les clés KMS de chiffrement symétrique	✓	⊗

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
GenerateDataKeyPairWithoutPlaintext	✓ Valable uniquement sur les clés KMS de chiffrement symétrique	✓	⊗
GenerateDataKeyWithoutPlaintext	✓ Valable uniquement sur les clés KMS de chiffrement symétrique	✓	✓
GenerateMac Valable uniquement sur les clés KMS HMAC	✓	✓	⊗
GetKeyPolicy	✓	✓	✓
GetKeyRotationStatus	✓	✓ (`KeyRotationEnabled` sera toujours `false`.)	⊗

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
GetParametersForImport	✓ Valable uniquement pour les clés avec éléments de clé importés (`Origin` est `EXTERNAL`)	✓	⊗
GetPublicKey Valable uniquement pour les clés KMS asymétriques	✓	✓	⊗
ImportKeyMaterial	✓ Valable uniquement pour les clés avec éléments de clé importés (`Origin` est `EXTERNAL`)	✓	⊗
ListAliases	✓	✓	✓
ListGrants	✓	✓	✓
ListKeyPolicies	✓	✓	✓

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
ListKeyRotations	✓ Valable uniquement sur les clés de chiffrement symétriques avec `AWS_KMS` origine.	✓ Valable uniquement sur les clés de chiffrement symétriques à région unique.	⊗
ListResourceTags	✓	✓	✓
ListRetirableGrants	✓	✓	✓
PutKeyPolicy	✓	✓	✓
ReEncrypt	✓ Valable uniquement lorsque `KeyUsage` est `ENCRYPT_DECRYPT`	✓	✓

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
ReplicateKey	✓ Valable uniquement sur les clés primaires multi-régions	✓ Valable uniquement sur les clés primaires multi-régions	⊗
RetireGrant	✓	✓	✓
RevokeGrant	✓	✓	✓
RotateKeyOnDemand	✓ Valable uniquement sur les clés de chiffrement symétriques avec `AWS_KMS` origine.	✓ Valable uniquement sur les clés de chiffrement symétriques à région unique.	⊗
ScheduleKeyDeletion	✓	✓	✓
Sign (Signer) Valable uniquement lorsque `KeyUsage` est `SIGN_VERIFY`	✓	✓	⊗
TagResource	✓	✓	✓

AWS KMS Fonctionnement de l'API	Clés multi-région	Éléments de clé importés	Clés KMS dans un magasin de clés personnalisé
UntagResource	✓	✓	✓
UpdateAlias – La clé KMS actuelle et la nouvelle clé KMS doivent être du même type (toutes deux soit symétriques, soit asymétriques, soit HMAC) et avoir la même utilisation de clé.	✓	✓	✓
UpdateKeyDescription	✓	✓	✓
UpdateReplicaRegion	✓	✓ Valable uniquement sur les clés multi-régions	⊗
Vérification Valide uniquement lorsque `KeyUsage` est `SIGN_VERIFY` .	✓	✓	⊗
VerifyMac Valable uniquement sur les clés KMS HMAC	✓	✓	⊗

Référence de spécification clé

Lorsque vous créez une clé KMS asymétrique ou une clé KMS HMAC, vous sélectionnez sa spécification de clé. La spécification de clé, qui est une propriété de Every AWS KMS key, représente la configuration cryptographique de votre clé KMS. Vous choisissez la spécification de clé lorsque vous créez la clé KMS et vous ne pouvez pas la modifier. Si vous avez sélectionné la mauvaise spécification de clé, supprimez la clé KMS et créez-en une autre.

> **Note**
>
> La spécification de clé d'une clé KMS était appelée « spécification de la clé principale du client ». Le `CustomerMasterKeySpec` paramètre de l'CreateKeyopération est obsolète. Utilisez plutôt le paramètre `KeySpec`. La réponse des DescribeKeyopérations `CreateKey` et inclut un `CustomerMasterKeySpec` membre `KeySpec` et ayant la même valeur.

La spécification de clé détermine si la clé KMS est symétrique ou asymétrique, le type de contenu clé de la clé KMS et les algorithmes de chiffrement, de signature ou de code d'authentification des messages (MAC) compatibles AWS KMS avec la clé KMS. La spécification de clé que vous choisissez est généralement déterminée par votre cas d'utilisation et vos exigences réglementaires. Cela dit, les opérations de chiffrement sur des clés KMS dont les spécifications sont différentes sont soumises à des tarifs et à des quotas différents. Pour plus d'informations sur la tarification, consultez la page AWS Key Management Service Pricing (Tarification). Pour de plus amples informations sur les quotas de demande, veuillez consulter Quotas de demande.

Pour limiter les spécifications clés que les principaux peuvent utiliser lors de la création de clés KMS, utilisez la clé de KeySpec condition kms :. Vous pouvez également utiliser la clé de `kms:KeySpec` condition pour autoriser les principaux à appeler des AWS KMS opérations uniquement sur des clés KMS avec une spécification de clé particulière. Par exemple, vous pouvez rejeter l'autorisation de planifier la suppression d'une clé KMS avec une spécification de clé `RSA_4096`.

AWS KMS prend en charge les spécifications clés suivantes pour les clés KMS :

Spécifications de clé de chiffrement symétrique (par défaut)

- SYMMETRIC_DEFAULT

Spécifications de clés RSA (chiffrement et déchiffrement, ou signature et vérification)

- RSA_2048

- RSA_3072
- RSA_4096

Spécifications de la clé de courbe elliptique

- Paires de clés asymétriques à courbe elliptique recommandées par le NIST (signature et vérification, ou obtention de secrets partagés)
  - ECC_NIST_P256 (secp256r1)
  - ECC_NIST_P384 (secp384r1)
  - ECC_NIST_P521 (secp521r1)
- Autres paires de clés asymétriques de courbe elliptique (signature et vérification)
  - ECC_SECG_P256K1 (secp256k1), couramment utilisé pour la crypto-monnaie.

SM2 spécification de la clé (chiffrement et déchiffrement, ou signature et vérification, ou obtention de secrets partagés)

- SM2 (Régions de Chine uniquement)

Spécifications de clé HMAC

- HMAC_224
- HMAC_256
- HMAC_384
- HMAC_512

Caractéristiques clés de la ML-DSA

- ML_DSA_44
- ML_DSA_65
- ML_DSA_87

Spécification de clé SYMMETRIC_DEFAULT

La spécification de clé par défaut, SYMMETRIC_DEFAULT, est la spécification de clé pour les clés KMS de chiffrement symétriques. Lorsque vous sélectionnez le type de clé symétrique et l'utilisation de la clé de chiffrement et de déchiffrement dans la AWS KMS console, la spécification de la `SYMMETRIC_DEFAULT` clé est sélectionnée. Dans l'CreateKeyopération, si vous ne spécifiez aucune `KeySpec` valeur, SYMMETRIC_DEFAULT est sélectionné. Si vous n'avez pas de raison d'utiliser une spécification de clé différente, SYMMETRIC_DEFAULT est un bon choix.

SYMMETRIC_DEFAULT représente AES-256-GCM, un algorithme symétrique basé sur la norme de chiffrement avancée (AES) en mode compteur Galois (GCM) avec des clés de 256 bits, une norme industrielle pour le chiffrement sécurisé. Le texte chiffré généré par cet algorithme prend en charge les données authentifiées supplémentaires (AAD), telles qu'un contexte de chiffrement et GCM fournit une vérification d'intégrité supplémentaire sur le texte chiffré.

Les données chiffrées sous AES-256-GCM sont protégées maintenant et à l'avenir. Les cryptographes considèrent cet algorithme comme résistant quantique. Dans un avenir théorique, les attaques de calcul quantique à grande échelle sur les textes chiffrés créés sous les clés AES-GCM 256 bits réduiront la sécurité effective de la clé à 128 bits. Mais ce niveau de sécurité est suffisant pour empêcher les attaques par force brute sur des AWS KMS textes chiffrés.

Seule exception dans les régions de Chine, où SYMMETRIC_DEFAULT représente une clé symétrique de 128 bits qui utilise le chiffrement. SM4 Vous ne pouvez créer une SM4 clé de 128 bits que dans les régions de Chine. Vous ne pouvez pas créer une clé AES-GCM KMS AES-GCM 256 bits dans les régions de Chine.

Vous pouvez utiliser une clé KMS de chiffrement symétrique AWS KMS pour chiffrer, déchiffrer et rechiffrer les données, ainsi que pour protéger les clés de données et les paires de clés de données générées. AWS les services intégrés AWS KMS utilisent des clés KMS de chiffrement symétriques pour chiffrer vos données au repos. Vous pouvez importer vos propres éléments de clé dans une clé KMS de chiffrement symétrique et créer des clés KMS de chiffrement symétriques dans des magasins de clés personnalisés. Pour obtenir un tableau comparant les opérations que vous pouvez effectuer sur les clés KMS symétriques et asymétriques, veuillez consulter la Comparaison des clés KMS symétriques et asymétriques.

Vous pouvez utiliser une clé KMS de chiffrement symétrique AWS KMS pour chiffrer, déchiffrer et rechiffrer les données, ainsi que pour générer des clés de données et des paires de clés de données. Vous pouvez créer des clés KMS de chiffrement symétriques multi-région, importer vos propres éléments de clé vers une clé KMS de chiffrement symétrique et créer des clés KMS de chiffrement symétriques dans des magasins de clés personnalisés. Pour obtenir un tableau de comparaison des opérations que vous pouvez exécuter sur des clés KMS de différents types, veuillez consulter.Référence des types de clés.

Spécifications de clés RSA

Lorsque vous utilisez une spécification de clé RSA, AWS KMS crée une clé KMS asymétrique avec une paire de clés RSA. La clé privée ne sort jamais AWS KMS non chiffrée. Vous pouvez utiliser la

clé publique contenue dans le AWS KMS fichier ou télécharger la clé publique pour une utilisation en dehors de celui-ci AWS KMS.

> **Warning**
>
> Lorsque vous chiffrez des données en dehors de AWS KMS, assurez-vous de pouvoir déchiffrer votre texte chiffré. Si vous utilisez la clé publique d'une clé KMS qui a été supprimée de AWS KMS, la clé publique d'une clé KMS configurée pour la signature et la vérification, ou un algorithme de chiffrement qui n'est pas pris en charge par la clé KMS, les données seront irrécupérables.

Dans AWS KMS, vous pouvez utiliser des clés KMS asymétriques avec des paires de clés RSA pour le chiffrement et le déchiffrement, ou pour la signature et la vérification, mais pas les deux. Cette propriété, appelée Key usage (utilisation de la clé), est déterminée en marge des spécifications de la clé, mais vous devez prendre cette décision avant de sélectionner une spécification de clé.

AWS KMS prend en charge les spécifications clés RSA suivantes pour le chiffrement et le déchiffrement ou pour la signature et la vérification :

- RSA_2048
- RSA_3072
- RSA_4096

Les spécifications des clés RSA diffèrent par la longueur de la clé RSA en bits. La spécification de la clé RSA que vous choisissez peut être déterminée par vos normes de sécurité ou les exigences de votre tâche. En général, utilisez la plus grande clé qui est pratique et abordable pour votre tâche. Les opérations de chiffrement sur des clés KMS dont les spécifications de clés RSA sont différentes sont soumises à des tarifs différents. Pour plus d'informations sur la AWS KMS tarification, consultez la section Tarification des services de gestion des AWS clés. Pour de plus amples informations sur les quotas de demande, veuillez consulter Quotas de demande.

Spécifications des clés RSA pour le chiffrement et le déchiffrement

Lorsqu'une clé KMS asymétrique RSA est utilisée pour le chiffrement et le déchiffrement, vous chiffrez avec la clé publique et déchiffrez avec la clé privée. Lorsque vous appelez l'`Encrypt`opération AWS KMS pour obtenir une clé RSA KMS, AWS KMS utilise la clé publique de

la paire de clés RSA et l'algorithme de chiffrement que vous spécifiez pour chiffrer vos données. Pour déchiffrer le texte chiffré, appelez l'`Decrypt`opération et spécifiez la même clé KMS et le même algorithme de chiffrement. AWS KMS utilise ensuite la clé privée de la paire de clés RSA pour déchiffrer vos données.

Vous pouvez également télécharger la clé publique et l'utiliser pour chiffrer des données en dehors de AWS KMS. Veillez à utiliser un algorithme de chiffrement AWS KMS compatible avec les clés RSA KMS. Pour déchiffrer le texte chiffré, appelez la fonction `Decrypt` avec les mêmes clé KMS et algorithme de chiffrement.

AWS KMS prend en charge deux algorithmes de chiffrement pour les clés KMS avec des spécifications de clé RSA. Ces algorithmes, définis dans PKCS #1 v2.2, diffèrent par la fonction de hachage qu'ils utilisent en interne. Dans AWS KMS, les algorithmes RSAES_OAEP utilisent toujours la même fonction de hachage à des fins de hachage et pour la fonction de génération de masque (). MGF1 Vous devez spécifier un algorithme de chiffrement lorsque vous appelez les opérations Encrypt (Chiffrer) et Decrypt (Déchiffrer). Vous pouvez choisir un algorithme différent pour chaque requête.

Algorithmes de chiffrement pris en charge pour les spécifications de clé RSA

Algorithme de chiffrement	Description de l'algorithme
RSAES_OAEP_SHA_1	PKCS #1 v2.2, section 7.1. Chiffrement RSA avec rembourrage OAEP utilisant SHA-1 à la fois pour le hachage et la fonction de générati on de MGF1 masque, avec une étiquette vide.
RSAES_OAEP_SHA_256	PKCS #1, section 7.1. Chiffrement RSA avec rembourrage OAEP utilisant SHA-256 à la fois pour le hachage et la fonction de génération de MGF1 masque, avec une étiquette vide.

Vous ne pouvez pas configurer une clé KMS pour utiliser un algorithme de chiffrement particulier. Cependant, vous pouvez utiliser la condition kms : EncryptionAlgorithm policy pour spécifier les algorithmes de chiffrement que les principaux sont autorisés à utiliser avec la clé KMS.

Pour obtenir les algorithmes de chiffrement d'une clé KMS, consultez la configuration cryptographique de la clé KMS dans la AWS KMS console ou utilisez l'DescribeKeyopération.

AWS KMS fournit également les spécifications de clé et les algorithmes de chiffrement lorsque vous téléchargez votre clé publique, soit dans la AWS KMS console, soit à l'aide de l'GetPublicKeyopération.

Vous pouvez choisir une spécification de clé RSA en fonction de la longueur des données en texte brut que vous pouvez chiffrer dans chaque requête. Le tableau suivant indique la taille maximale, en octets, du texte brut que vous pouvez chiffrer en un seul appel à l'opération Encrypt (chiffrer). Les valeurs diffèrent selon la spécification de la clé et l'algorithme de chiffrement. Pour comparer, vous pouvez utiliser une clé KMS de chiffrement symétrique pour chiffrer jusqu'à 4 096 octets en même temps.

Pour calculer la longueur maximale du texte brut en octets pour ces algorithmes, utilisez la formule suivante : (*key_size_in_bits*/8) - (2 * *hash_length_in_bits* /8) - 2. Par exemple, pour RSA_2048 avec SHA-256, la taille maximale du texte brut en octets est (2048/8) - (2 * 256/8) -2 = 190.

Taille maximale du texte brut (en octets) dans une opération de chiffrement

	Algorithme de chiffrement	
Spécifications de la clé	RSAES_OAEP_SHA_1	RSAES_OAEP_SHA_256
RSA_2048	214	190
RSA_3072	342	318
RSA_4096	470	446

Spécifications des clés RSA pour la signature et la vérification

Lorsqu'une clé KMS asymétrique RSA est utilisée pour la signature et la vérification, vous générez la signature d'un message avec la clé privée et vérifiez une signature avec la clé publique.

Lorsque vous appelez l'`Sign`opération AWS KMS pour une clé KMS asymétrique, AWS KMS utilise la clé privée de la paire de clés RSA, le message et l'algorithme de signature que vous spécifiez pour générer une signature. Pour vérifier la signature, appelez l'opération Vérifier. Spécifiez la signature, ainsi que la même clé KMS, le même message et le même algorithme de signature. AWS KMS utilise ensuite la clé publique de la paire de clés RSA pour vérifier la signature. Vous pouvez également télécharger la clé publique et l'utiliser pour vérifier la signature en dehors de AWS KMS.

AWS KMS prend en charge les algorithmes de signature suivants pour toutes les clés KMS avec une spécification de clé RSA. Vous devez spécifier un algorithme de signature lorsque vous appelez les opérations Sign (Signer) et Verify (Vérifier). Vous pouvez choisir un algorithme différent pour chaque requête. Lors de la signature avec des paires de clés RSA, les algorithmes RSASSA-PSS sont privilégiés. Nous incluons les algorithmes RSASSA- PKCS1 -v1_5 pour assurer la compatibilité avec les applications existantes.

Algorithmes de signature pris en charge pour les spécifications de clés RSA

Algorithme de signature	Description de l'algorithme
RSASSA_PSS_SHA_256	PKCS #1 v2.2, Section 8.1, signature RSA avec rembourrage PSS utilisant SHA-256 pour le résumé du message et la fonction de génération de MGF1 masque, ainsi qu'un sel de 256 bits
RSASSA_PSS_SHA_384	PKCS #1 v2.2, Section 8.1, signature RSA avec rembourrage PSS utilisant SHA-384 pour le résumé du message et la fonction de génération de MGF1 masque, ainsi qu'un sel de 384 bits
RSASSA_PSS_SHA_512	PKCS #1 v2.2, Section 8.1, signature RSA avec rembourrage PSS utilisant SHA-512 pour le résumé du message et la fonction de génération de MGF1 masque, ainsi qu'un sel de 512 bits
RSASSA_ _V1_5_SHA_256 PKCS1	PKCS #1 v2.2, section 8.2, signature RSA avec remplissage PKCS #1v1.5 et SHA-256
RSASSA_ _V1_5_SHA_384 PKCS1	PKCS #1 v2.2, section 8.2, signature RSA avec remplissage PKCS #1v1.5 et SHA-384
RSASSA_ _V1_5_SHA_512 PKCS1	PKCS #1 v2.2, section 8.2, signature RSA avec remplissage PKCS #1v1.5 et SHA-512

Vous ne pouvez pas configurer une clé KMS pour utiliser des algorithmes de signature particuliers. Cependant, vous pouvez utiliser la condition kms : SigningAlgorithm policy pour spécifier les algorithmes de signature que les principaux sont autorisés à utiliser avec la clé KMS.

Pour obtenir les algorithmes de signature d'une clé KMS, consultez la configuration cryptographique de la clé KMS dans la AWS KMS console ou en utilisant l'DescribeKeyopération. AWS KMS fournit également les spécifications clés et les algorithmes de signature lorsque vous téléchargez votre clé publique, soit dans la AWS KMS console, soit à l'aide de l'GetPublicKeyopération.

Spécifications de la clé de courbe elliptique

Lorsque vous utilisez une spécification de clé à courbe elliptique (ECC), vous AWS KMS créez une clé KMS asymétrique avec une paire de clés ECC pour la signature et la vérification ou pour obtenir des secrets partagés (mais pas les deux). La clé privée qui génère des signatures ou déduit des secrets partagés n'est jamais AWS KMS déchiffrée. Vous pouvez utiliser la clé publique pour vérifier les signatures internes AWS KMS ou télécharger la clé publique pour une utilisation en dehors de AWS KMS.

AWS KMS prend en charge les spécifications clés ECC suivantes pour les clés KMS asymétriques.

- Paires de clés asymétriques à courbe elliptique recommandées par le NIST (signature et vérification, ou obtention de secrets partagés)
  - ECC_NIST_P256 (secp256r1)
  - ECC_NIST_P384 (secp384r1)
  - ECC_NIST_P521 (secp521r1)
- Autres paires de clés asymétriques de courbe elliptique (signature et vérification)
  - ECC_SECG_P256K1 (secp256k1), couramment utilisé pour les crypto-monnaies.

La spécification de clé ECC que vous choisissez peut être déterminée par vos normes de sécurité ou les exigences de votre tâche. En général, utilisez la courbe qui est la plus pratique et abordable pour votre tâche.

Si vous créez une clé KMS asymétrique pour obtenir des secrets partagés, utilisez l'une des spécifications de clé à courbe elliptique recommandées par le NIST. Le seul algorithme d'accord de clés pris en charge pour dériver des secrets partagés est le cofacteur Diffie-Hellman (Elliptic Curve Cryptography Cofactor Diffie-Hellman Primitive). Pour un exemple de dérivation de secrets partagés hors ligne, consultezthe section called “Découverte de secrets partagés hors ligne”.

Si vous créez une clé KMS asymétrique à utiliser avec les crypto-monnaies, utilisez la spécification de clé ECC_SECG_P256K1. Vous pouvez également utiliser cette spécification de clé à d'autres fins, mais elle est nécessaire pour Bitcoin et d'autres crypto-monnaies.

Les clés KMS avec des spécifications de clés ECC différentes sont tarifées différemment et sont soumises à des quotas de demande différents. Pour plus d'informations sur la AWS KMS tarification, consultez la section AWS Key Management Service Tarification. Pour de plus amples informations sur les quotas de demande, veuillez consulter Quotas de demande.

Le tableau suivant présente les algorithmes de signature compatibles AWS KMS avec chacune des spécifications clés de l'ECC. Vous ne pouvez pas configurer une clé KMS pour utiliser des algorithmes de signature particuliers. Cependant, vous pouvez utiliser la condition kms : SigningAlgorithm policy pour spécifier les algorithmes de signature que les principaux sont autorisés à utiliser avec la clé KMS.

Algorithmes de signature pris en charge pour les spécifications de clés ECC

Spécifications de la clé	Algorithme de signature	Description de l'algorithme
ECC_NIST_P256	ECDSA_SHA_256	NIST FIPS 186-4, Section 6.4, ECDSA signature using the curve specified by the key and SHA-256 pour le résumé du message.
ECC_NIST_P384	ECDSA_SHA_384	NIST FIPS 186-4, Section 6.4, ECDSA signature using the curve specified by the key and SHA-384 pour le résumé du message.
ECC_NIST_P521	ECDSA_SHA_512	NIST FIPS 186-4, Section 6.4, ECDSA signature using the curve specified by the key and SHA-512 pour le résumé du message.
ECC_SECG_P256K1	ECDSA_SHA_256	NIST FIPS 186-4, Section 6.4, ECDSA signature using the

Spécifications de la clé	Algorithme de signature	Description de l'algorithme
		curve specified by the key and SHA-256 pour le résumé du message.

Spécifications de clé pour les clés KMS HMAC

AWS KMS prend en charge les clés HMAC symétriques de différentes longueurs. La spécification de clé que vous sélectionnez peut dépendre de vos exigences de sécurité, réglementaires ou métier. La longueur de la clé détermine l'algorithme MAC utilisé dans les VerifyMacopérations GenerateMacet. En général, les clés plus longues sont plus sécurisées. Utilisez la clé la plus longue qui est pratique pour votre cas d'utilisation.

Spécification de clé HMAC	Algorithme MAC
HMAC_224	HMAC_SHA_224
HMAC_256	HMAC_SHA_256
HMAC_384	HMAC_SHA_384
HMAC_512	HMAC_SHA_512

Caractéristiques clés de la ML-DSA

Une clé ML-DSA est une clé cryptographique utilisée dans l'algorithme de signature Module-Lattice-Based numérique (ML-DSA), conçu pour la cryptographie post-quantique. Cet algorithme fait partie des efforts de normalisation du NIST (National Institute of Standards and Technology), spécifiquement décrits dans les normes fédérales de traitement de l'information (FIPS) 204.

Les clés ML-DSA sont utilisées dans un système de paires de clés public-privé. La clé privée est utilisée pour signer les données, tandis que la clé publique est utilisée pour vérifier la signature. Ce système garantit l'authenticité, l'intégrité et la non-répudiation des messages ou documents numériques, même en cas de menaces informatiques quantiques potentielles.

Lorsque vous créez une clé avec la spécification de clé ML-DSA, vous AWS KMS créez une clé KMS asymétrique avec une paire de clés ML-DSA pour la signature et la vérification. La clé privée qui

génère les signatures n'est jamais AWS KMS déchiffrée. Vous pouvez utiliser la clé publique pour vérifier les signatures internes AWS KMS ou télécharger la clé publique pour une utilisation en dehors de AWS KMS.

AWS KMS prend en charge les spécifications clés ML-DSA suivantes pour les clés KMS asymétriques :

- ML_DSA_44
- ML_DSA_65
- ML_DSA_87

AWS KMS prend en charge l'algorithme de signature ML_DSA_SHAKE_256 pour toutes les spécifications clés ML-DSA.

SM2 spécification clé (régions de Chine uniquement)

La spécification SM2 clé est une spécification clé à courbe elliptique définie dans la GM/T série de spécifications publiées par le Bureau de l'administration nationale de cryptographie commerciale (OSCCA) de la Chine. La spécification SM2 clé n'est disponible que dans les régions de Chine. Lorsque vous utilisez la spécification de SM2 clé, AWS KMS crée une clé KMS asymétrique avec une paire de SM2 clés. Vous pouvez utiliser votre paire de SM2 clés à l'intérieur AWS KMS ou télécharger la clé publique pour une utilisation en dehors de AWS KMS. Pour de plus amples informations, veuillez consulter the section called “Vérification hors ligne à l'aide de paires de SM2 clés (régions de Chine uniquement)”.

Chaque clé KMS ne peut avoir qu'un seul type d'utilisation de clé. Vous pouvez utiliser une clé SM2 KMS pour la signature et la vérification, le chiffrement et le déchiffrement, ou pour obtenir des secrets partagés. Vous devez spécifier l’utilisation de la clé lorsque vous créez la clé KMS, et vous ne pouvez pas la modifier une fois la clé créée.

Si vous créez une clé KMS asymétrique pour obtenir des secrets partagés, utilisez la spécification de SM2 clé. Le seul algorithme d'accord de clés pris en charge pour dériver des secrets partagés est le cofacteur Diffie-Hellman (Elliptic Curve Cryptography Cofactor Diffie-Hellman Primitive).

AWS KMS prend en charge les algorithmes SM2 de chiffrement et de signature suivants :

- SM2Algorithme de chiffrement PKE

  SM2Le PKE est un algorithme de chiffrement basé sur une courbe elliptique défini par l'OSCCA en 0003.4-2012. GM/T

- SM2Algorithme de signature DSA

  SM2Le DSA est un algorithme de signature basé sur une courbe elliptique défini par l'OSCCA en 0003.2-2012. GM/T SM2Le DSA nécessite un identifiant distinctif haché à l'aide de l'algorithme de SM3 hachage, puis combiné avec le message, ou le résumé du message, auquel vous avez transmis le message. AWS KMS Cette valeur concaténée est ensuite hachée et signée par. AWS KMS

AWS KMS autorisations

Ce tableau est conçu pour vous aider à comprendre AWS KMS les autorisations afin que vous puissiez contrôler l'accès à vos AWS KMS ressources. Les définitions des en-têtes de colonne apparaissent sous le tableau.

Vous pouvez également en savoir plus sur AWS KMS les autorisations dans la AWS Key Management Service rubrique Actions, ressources et clés de condition de la référence d'autorisation de service. Toutefois, cette rubrique ne répertorie pas toutes les clés de condition que vous pouvez utiliser pour affiner chaque autorisation.

Pour plus d'informations sur les AWS KMS opérations valides pour les clés KMS de chiffrement symétriques, les clés KMS asymétriques et les clés KMS HMAC, consultez le. Référence des types de clés

> **Note**
>
> Vous devrez peut-être faire défiler horizontalement ou verticalement pour voir toutes les données de la table.

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
CancelKeyDeletion `kms:CancelKeyDeletion`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
ConnectCustomKeyStore `kms:ConnectCustomKeyStore`	Politique IAM	Non	*	km : CallerAccount
CreateAlias `kms:CreateAlias` Pour utiliser cette opération, l'appelant doit avoir l'autorisation	Politique IAM (pour l'alias)	Non	Alias	Aucune (lorsque vous contrôlez l'accès à l'alias)
	Politique de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
`kms:CreateAlias` sur deux ressources : • L'alias (dans une politique IAM) • La clé KMS (dans une politique de clé) Pour plus de détails, veuillez consulter Contrôle de l'accès aux alias.	(pour la clé KMS)			km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
CreateCustomKeyStore `kms:CreateCustomKeyStore`	Politique IAM	Non	*	km : CallerAccount

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
CreateGrant `kms:CreateGrant`	Stratégie de clé	Oui	Clé KMS	Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions d'octroi : km : GrantConstraintType km : GranteePrincipal km : GrantIsForAWSResource km : GrantOperations km : RetiringPrincipal Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
CreateKey `kms:CreateKey`	Politique IAM	Non	*	km : BypassPolicyLockoutSafetyCheck km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ViaService aws :RequestTag/tag-key (clé de condition AWS globale) aws :ResourceTag/tag-key (clé de condition AWS globale) aws : TagKeys (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
Decrypt `kms:Decrypt`	Politique de clé	Oui	Clé KMS	Conditions des opérations de chiffrement km : EncryptionAlgorithm km : RequestAlias Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ViaService
DeleteAlias `kms:DeleteAlias` Pour utiliser cette opération, l'appelant doit avoir l'autorisation `kms:DeleteAlias` sur deux ressources : • L'alias (dans une politique IAM) • La clé KMS (dans une politique de clé) Pour plus de détails, veuillez consulter Contrôle de l'accès aux alias.	Politique IAM (pour l'alias)	Non	Alias	Aucune (lorsque vous contrôlez l'accès à l'alias)
	Politique de clé (pour la clé KMS)	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
DeleteCustomKeyStore `kms:DeleteCustomKeyStore`	Politique IAM	Non	*	km : CallerAccount

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
DeleteImportedKeyMaterial `kms:DeleteImportedKeyMaterial`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
DeriveSharedSecret `kms:DeriveSharedSecret`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Conditions des opérations de chiffrement : km : KeyAgreementAlgorithm
DescribeCustomKeyStores `kms:DescribeCustomKeyStores`	Politique IAM	Non	*	km : CallerAccount

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
DescribeKey `kms:DescribeKey`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : RequestAlias

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
DisableKey `kms:DisableKey`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
DisableKeyRotation `kms:DisableKeyRotation`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
DisconnectCustomKeyStore `kms:DisconnectCustomKeyStore`	Politique IAM	Non	*	km : CallerAccount

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
EnableKey `kms:EnableKey`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
EnableKeyRotation `kms:EnableKeyRotation`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Conditions de rotation automatique des touches : km : RotationPeriodInDays

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
Encrypt `kms:Encrypt`	Politique de clé	Oui	Clé KMS	Conditions des opérations de chiffrement km : EncryptionAlgorithm km : RequestAlias Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GenerateDataKey `kms:GenerateDataKey`	Stratégie de clé	Oui	Clé KMS	Conditions des opérations de chiffrement km : EncryptionAlgorithm km : RequestAlias Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GenerateDataKeyPair `kms:GenerateDataKeyPair`	Stratégie de clé	Oui	Clé KMS Génère une paire de clés de données asymétriques qui est protégée par une clé KMS de chiffrement symétrique.	Conditions pour les paires de clés de données : km : DataKeyPairSpec Conditions des opérations de chiffrement km : EncryptionAlgorithm km : RequestAlias Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GenerateDataKeyPairWithoutPlaintext `kms:GenerateDataKeyPairWithoutPlaintext`	Stratégie de clé	Oui	Clé KMS Génère une paire de clés de données asymétriques qui est protégée par une clé KMS de chiffrement symétrique.	Conditions pour les paires de clés de données : km : DataKeyPairSpec Conditions des opérations de chiffrement km : EncryptionAlgorithm km : RequestAlias Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GenerateDataKeyWithoutPlaintext `kms:GenerateDataKeyWithoutPlaintext`	Stratégie de clé	Oui	Clé KMS	Conditions des opérations de chiffrement km : EncryptionAlgorithm km : RequestAlias Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ViaService
GenerateMac kms:GenerateMac	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Conditions des opérations de chiffrement : km : MacAlgorithm km : RequestAlias
GenerateRandom kms:GenerateRandom	Politique IAM	N/A	*	Aucun

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GetKeyPolicy `kms:GetKeyPolicy`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GetKeyRotationStatus `kms:GetKeyRotationStatus`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GetParametersForImport `kms:GetParametersForImport`	Stratégie de clé	Non	Clé KMS	km : WrappingAlgorithm km : WrappingKeySpec Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
GetPublicKey `kms:GetPublicKey`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : RequestAlias

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ImportKeyMaterial `kms:ImportKeyMaterial`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : ExpirationModel km : ValidTo
ListAliases `kms:ListAliases`	Politique IAM	Non	*	Aucun

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ListGrants `kms:ListGrants`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : GrantIsForAWSResource

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ListKeyPolicies `kms:ListKeyPolicies`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ListKeyRotations `kms:ListKeyRotations`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
ListKeys `kms:ListKeys`	Politique IAM	Non	*	Aucun

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ListResourceTags `kms:ListResourceTags`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ListRetirableGrants `kms:ListRetirableGrants`	Politique IAM	Le principal spécifié doit être dans le compte local, mais l'opération renvoie des octrois dans tous les comptes.	*	Aucun

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
PutKeyPolicy `kms:PutKeyPolicy`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : BypassPolicyLockoutSafetyCheck

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition

ReEncrypt
 	
`kms:ReEncryptFrom`
 	
`kms:ReEncryptTo`
 	
Pour utiliser cette opération, l'appelant doit avoir l'autorisation sur deux clés KMS :
 	
• `kms:ReEncryptFrom` sur la clé KMS utilisée pour déchiffrer
• `kms:ReEncryptTo` sur la clé KMS utilisée pour chiffrer
	Politique de clé	Oui	Clé KMS	Conditions des opérations de chiffrement
				
				km : EncryptionAlgorithm
				
				km : RequestAlias
				
				Conditions du contexte de chiffrement :
				
				kms EncryptionContext : touche contextuelle
				
				km : EncryptionContextKeys
				
				Conditions pour les opérations de clé KMS :
				
				km : CallerAccount
				
				km : KeySpec
				
				km : KeyUsage
				
				km : KeyOrigin
				
				km : MultiRegion
				
				km : MultiRegionKeyType
				
				km : ResourceAliases
				
				aws :ResourceTag/tag-key (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
				km : ViaService Autres conditions : km : ReEncryptOnSameKey
ReplicateKey `kms:ReplicateKey` Pour utiliser cette opération, l'appelant doit avoir les autorisations suivantes : • `kms:ReplicateKey` sur la clé principale multi-région • `kms:CreateKey` dans une politique IAM dans la région de réplica	Politique de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : ReplicaRegion

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
RetireGrant `kms:RetireGrant` L'autorisation de retirer un octroi est déterminée principalement par l'octroi. Une politique seule ne peut pas autoriser l'accès à cette opération. Pour de plus amples informations, veuillez consulter Retrait et révocation d'octrois.	Politique IAM (Cette autorisation n'est pas effective dans une politique de clé.)	Oui	Clé KMS	Conditions du contexte de chiffrement : kms EncryptionContext : touche contextuelle km : EncryptionContextKeys Conditions d'octroi : km : GrantConstraintType Conditions pour les opérations de clé KMS : km : CallerAccount km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
RevokeGrant `kms:RevokeGrant`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions : km : GrantIsFor AWSResource

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
RotateKeyOnDemand `kms:RotateKeyOnDemand`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
ScheduleKeyDeletion `kms:ScheduleKeyDeletion`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
Sign (Signer) `kms:Sign`	Politique de clé	Oui	Clé KMS	Conditions de signature et de vérification : km : MessageType km : RequestAlias km : SigningAlgorithm Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
TagResource `kms:TagResource`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Conditions d'étiquetage : aws :RequestTag/tag-key (clé de condition AWS globale) aws : TagKeys (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
UntagResource `kms:UntagResource`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Conditions d'étiquetage : aws :RequestTag/tag-key (clé de condition AWS globale) aws : TagKeys (clé de condition AWS globale)

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
UpdateAlias `kms:UpdateAlias` Pour utiliser cette opération, l'appelant doit avoir l'autorisation `kms:UpdateAlias` sur trois ressources : • L'alias • La clé KMS actuellement associée • La clé KMS nouvellement associée Pour plus de détails, veuillez consulter Contrôle de l'accès aux alias.	Politique IAM (pour l'alias)	Non	Alias	Aucune (lorsque vous contrôlez l'accès à l'alias)
	Politique de clé (pour les clés KMS)	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService
UpdateCustomKeyStore `kms:UpdateCustomKeyStore`	Politique IAM	Non	*	km : CallerAccount

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
UpdateKeyDescription `kms:UpdateKeyDescription`	Stratégie de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
UpdatePrimaryRegion `kms:UpdatePrimaryRegion` Pour utiliser cette opération, l'appelant doit avoir l'autorisation `kms:UpdatePrimaryRegion` sur la clé principale multi-région qui deviendra une clé de réplica et sur la clé de réplica multi-région qui deviendra la clé principale.	Politique de clé	Non	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Autres conditions km : PrimaryRegion

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
Vérification `kms:Verify`	Politique de clé	Oui	Clé KMS	Conditions de signature et de vérification : km : MessageType km : RequestAlias km : SigningAlgorithm Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService

Actions et autorisations	Type de stratégie	Utilisation inter-comptes	Ressources (pour les politiques IAM)	AWS KMS clés de condition
VerifyMac `kms:VerifyMac`	Stratégie de clé	Oui	Clé KMS	Conditions pour les opérations de clé KMS : km : CallerAccount km : KeySpec km : KeyUsage km : KeyOrigin km : MultiRegion km : MultiRegionKeyType km : ResourceAliases aws :ResourceTag/tag-key (clé de condition AWS globale) km : ViaService Conditions des opérations de chiffrement : km : MacAlgorithm km : RequestAlias

Descriptions des colonnes

Les colonnes de ce tableau fournissent les informations suivantes :

- Actions et autorisations répertorie chaque opération AWS KMS d'API et l'autorisation qui autorise l'opération. Vous spécifiez l'opération dans l'élément `Action` d'une instruction de politique.

- Policy type (Type de politique) indique si l'autorisation peut être utilisée dans une politique de clé ou une politique IAM.

  Key policy (Politique de clé) signifie que vous pouvez spécifier l'autorisation dans la politique de clé. Lorsque la politique de clé contient l'instruction de politique qui active les politiques IAM, vous pouvez spécifier l'autorisation dans une politique IAM.

  IAM policy (Politique IAM) signifie que vous pouvez spécifier l'autorisation uniquement dans une politique IAM.

- Cross-account use (Utilisation inter-comptes) indique les opérations que les utilisateurs autorisés peuvent effectuer sur des ressources dans un autre Compte AWS.

  Une valeur de Yes (Oui) signifie que les principaux peuvent effectuer l'opération sur des ressources dans un autre Compte AWS.

  Une valeur de No (Non) signifie que les principaux peuvent effectuer l'opération uniquement sur des ressources dans leur propre Compte AWS.

  Si vous accordez à un principal dans un compte différent une autorisation qui ne peut pas être utilisée sur une ressource inter-comptes, l'autorisation n'est pas effective. Par exemple, si vous TagResource autorisez un mandant d'un autre compte KMS : à utiliser une clé KMS dans votre compte, ses tentatives de balisage de la clé KMS dans votre compte échoueront.

- Resources répertorie les AWS KMS ressources auxquelles les autorisations s'appliquent. AWS KMS prend en charge deux types de ressources : une clé KMS et un alias. Dans une politique de clé, la valeur de l'élément `Resource` est toujours `*`, ce qui indique la clé KMS à laquelle la politique de clé est attachée.

  Utilisez les valeurs suivantes pour représenter une AWS KMS ressource dans une politique IAM.

  Clé KMS

  Lorsque la ressource est une clé KMS, utilisez son ARN de clé. Pour obtenir de l'aide, veuillez consulter the section called “Trouvez l'ID et l'ARN de la clé”.

  `arn:`*`AWS_partition_name`*`:kms:`*`AWS_Region`*`:`*`AWS_account_ID`*`:key/`*`key_ID`*

  Par exemple :

  arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab

Alias

Lorsque la ressource est un alias, utilisez son ARN d'alias. Pour obtenir de l'aide, veuillez consulter the section called “Trouvez le nom de l'alias et l'ARN de l'alias”.

`arn:AWS_partition_name:kms:AWS_region:AWS_account_ID:alias/alias_name`

Par exemple :

arn:aws:kms:us-west- 2:111122223333 : alias/ ExampleAlias

***** (astérisque)

Lorsque l'autorisation ne s'applique pas à une ressource particulière (clé KMS ou alias), utilisez un astérisque (`*`).

Dans une politique IAM pour une AWS KMS autorisation, un astérisque dans l'`Resource`élément indique toutes les AWS KMS ressources (clés KMS et alias). Vous pouvez également utiliser un astérisque dans l'`Resource`élément lorsque l' AWS KMS autorisation ne s'applique à aucune clé ou alias KMS en particulier. Par exemple, lorsque vous autorisez `kms:CreateKey` ou refusez une `kms:ListKeys` autorisation, vous devez définir l'`Resource`élément sur`*`.

- AWS KMS les clés de AWS KMS condition répertorient les clés de condition que vous pouvez utiliser pour contrôler l'accès à l'opération. Vous spécifiez des conditions dans l'élément `Condition` d'une politique. Pour de plus amples informations, veuillez consulter AWS KMS clés de condition. Cette colonne inclut également les clés de condition AWS globales qui sont prises en charge par tous les AWS services AWS KMS, mais pas par tous.

AWS KMS opérations internes

AWS Key Management Service (AWS KMS) fournit des clés et des opérations cryptographiques sécurisées par des modules de sécurité matériels (HSM) validés par la norme de sécurité FIPS 140-3 de niveau 3 adaptés au cloud. AWS KMS les clés et les fonctionnalités sont utilisées par de nombreux services AWS cloud, et vous pouvez les utiliser pour protéger les données de vos applications. Ce guide technique fournit des détails sur les opérations cryptographiques qui sont exécutées AWS lorsque vous les utilisez AWS KMS.

AWS KMS des composants internes sont nécessaires pour faire évoluer et sécuriser un service de gestion HSMs des clés distribué dans le monde entier.

Rubriques

Domaines et état du domaine

Un ensemble coopératif d' AWS KMS entités internes fiables au sein d'un Région AWS est appelé domaine. Un domaine comprend un ensemble d'entités de confiance, un ensemble de règles et un ensemble de clés secrètes, appelées « clés de domaine ». Les clés de domaine sont partagées entre HSMs les membres du domaine. Un état de domaine se compose des champs suivants.

Nom

Un nom de domaine permettant d'identifier ce domaine.

Members

Une liste de ceux HSMs qui sont membres du domaine, y compris leur clé de signature publique et leurs clés d'accord public.

Opérateurs

Liste d'entités, de clés de signature publiques et d'un rôle (AWS KMS opérateur ou hôte du service) représentant les opérateurs de ce service.

Règles

Une liste des règles de quorum pour chaque commande qui doit être satisfaite pour exécuter une commande sur la clé HSM.

Clés de domaine

Une liste des clés de domaine (clés symétriques) actuellement utilisées dans le domaine.

L'état complet du domaine est uniquement disponible sur la clé HSM. L'état du domaine est synchronisé entre les membres du domaine HSM en tant que jeton de domaine exporté.

Clés de domaine

Tous les HSMs membres d'un domaine partagent un ensemble de clés de domaine, $\{DK_r\}$. Ces clés sont partagées via une routine d'exportation d'état de domaine. L'état de domaine exporté peut être importé dans n'importe quelle clé HSM membre du domaine.

L'ensemble des clés de domaine, $\{DK_r\}$, inclut toujours une clé de domaine active et plusieurs clés de domaine désactivées. Les clés de domaine font l'objet d'une rotation quotidienne afin de garantir AWS leur conformité à la Recommandation pour la gestion des clés - Partie 1. Pendant la rotation de la clé de domaine, toutes les clés KMS existantes chiffrées sous la clé de domaine sortante sont à nouveau chiffrées sous la nouvelle clé de domaine active. La clé de domaine active est utilisée pour chiffrer toute nouvelle EKTs clé. Les clés de domaine expirées ne peuvent être utilisées que pour déchiffrer le chiffrement précédemment chiffré EKTs pendant un nombre de jours équivalent au nombre de clés de domaine récemment modifiées.

Jetons de domaine exportés

Il existe un besoin régulier de synchroniser l'état entre les participants du domaine. Ceci est effectué en exportant l'état du domaine chaque fois qu'une modification est apportée au domaine. L'état du domaine est exporté en tant que jeton de domaine exporté.

Nom

Un nom de domaine permettant d'identifier ce domaine.

Members

Une liste de ceux HSMs qui sont membres du domaine, y compris leurs clés publiques de signature et d'accord.

Opérateurs

Une liste d'entités, de clés de signature publiques et d'un rôle qui représentent les opérateurs de ce service.

Règles

Une liste des règles de quorum pour chaque commande qui doit être satisfaite pour exécuter une commande sur un membre de domaine HSM.

Clés de domaine chiffrées

Clés de domaine chiffrées par enveloppe. Les clés de domaine sont chiffrées par le membre de signature pour chacun des membres mentionnés ci-dessus, enveloppées dans leur clé d'accord public.

Signature

Une signature sur l'état du domaine générée par une clé HSM, nécessairement membre du domaine qui a exporté l'état du domaine.

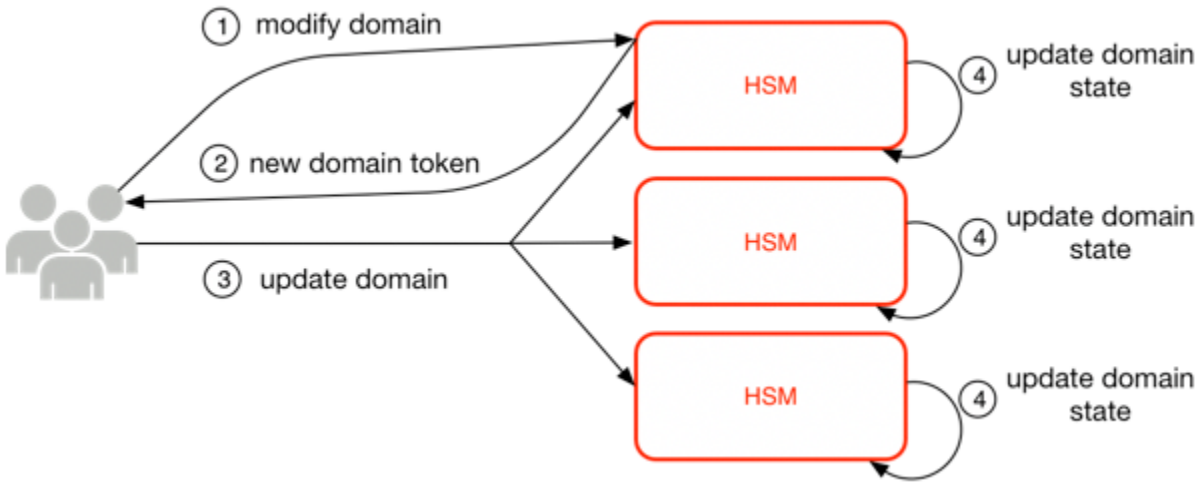
Le jeton de domaine exporté constitue la source essentielle de confiance pour les entités opérant dans le domaine.

Gestion des états de domaine

L'état du domaine est géré par des commandes authentifiées par quorum. Ces modifications comprennent la modification de la liste des participants de confiance dans le domaine, la modification des règles de quorum pour l'exécution des commandes HSM et la rotation régulière des clés de domaine. Ces commandes sont authentifiées sur une base individuelle, contrairement aux opérations de séance authentifiées ; comme illustré sur l’image suivante.

Dans son état initialisé et opérationnel, une HSM comporte un ensemble de clés d'identité asymétriques auto-générées, une paire de clés de signature et une paire de clés d'établissement de clé. Grâce à un processus manuel, un AWS KMS opérateur peut établir un domaine initial à créer sur un premier HSM dans une région. Ce domaine initial se compose d'un état de domaine complet, tel que défini précédemment dans cette rubrique. Il est installé via une commande jointe à chacun des membres HSM définis dans le domaine.

Après qu'une clé HSM ait rejoint un domaine initial, elle est associée aux règles qui sont définies dans ce domaine. Ces règles régissent les commandes qui utilisent les clés cryptographiques du client ou qui modifient l'état de l'hôte ou du domaine. Les opérations d'API de session authentifiées qui utilisent vos clés cryptographiques ont été définies précédemment.



L'image ci-dessus illustre la manière de modifier un état de domaine. Le processus se compose de quatre étapes :

1. Une commande basée sur le quorum est envoyée à une clé HSM pour modifier le domaine.
2. Un nouvel état de domaine est généré et exporté en tant que nouveau jeton de domaine exporté. L'état de la clé HSM n'est pas modifié, ce qui signifie que le changement n'est pas appliqué à la clé HSM.
3. Une deuxième commande est envoyée à chacun des membres du jeton HSMs de domaine nouvellement exporté pour mettre à jour l'état de leur domaine avec le nouveau jeton de domaine.
4. Les éléments HSMs répertoriés dans le nouveau jeton de domaine exporté peuvent authentifier la commande et le jeton de domaine. Ils peuvent également déballer les clés de domaine pour mettre à jour l'état du domaine sur tous HSMs les éléments du domaine.

HSMs ne communiquez pas directement les uns avec les autres. Au lieu de cela, un quorum d'opérateurs demande une modification de l'état du domaine, ce qui se traduit par un nouveau jeton de domaine exporté. Un hôte de service membre du domaine est utilisé pour distribuer le nouvel état de domaine à chaque clé HSM du domaine.

La sortie d'un domaine et l'entrée dans un domaine sont réalisées à l'aide des fonctions de gestion des clés HSM. La modification de l'état du domaine est réalisée à l'aide des fonctions de gestion du domaine.

Quitter un domaine

Permet à une clé HSM de quitter un domaine, en supprimant tous les restes et les clés de ce domaine de la mémoire.

Entrer dans un domaine

Permet à une clé HSM d'entrer dans un nouveau domaine ou de mettre à jour son état actuel de domaine vers le nouvel état de domaine. Le domaine existant est utilisé comme source de l'ensemble initial de règles pour authentifier ce message.

Créer un domaine

Provoque la création d'un nouveau domaine sur une clé HSM. Renvoie un premier jeton de domaine qui peut être distribué aux membres HSMs du domaine.

Modifier les opérateurs

Ajoute ou supprime des opérateurs de la liste des opérateurs autorisés et leurs rôles dans le domaine.

Modifier des membres

Ajoute ou supprime un HSM de la liste des utilisateurs autorisés HSMs dans le domaine.

Modifier les règles

Modifie l'ensemble des règles de quorum requises pour exécuter des commandes sur une clé HSM.

Rotation des clés de domaine

Permet de créer une nouvelle clé de domaine et de la marquer comme clé de domaine active. Cela déplace la clé active existante vers une clé désactivée et supprime la clé désactivée la plus ancienne de l'état du domaine.

Sécurité des communications internes

Les commandes entre les hôtes ou AWS KMS opérateurs du service et eux HSMs sont sécurisées par le biais de deux mécanismes décrits dans Sessions authentifiées : une méthode de demande signée par quorum et une session authentifiée utilisant un protocole hôte de service HSM.

Les commandes signées par quorum sont conçues de telle sorte qu'aucun opérateur ne puisse modifier les protections de sécurité critiques qu'elles fournissent. HSMs Les commandes qui s'exécutent sur les sessions authentifiées permettent de garantir que seuls les opérateurs de

service autorisés peuvent effectuer des opérations impliquant des clés KMS. Toutes les informations secrètes liées au client sont sécurisées dans l'ensemble de l' AWS infrastructure.

Établissement de clé

Pour sécuriser les communications internes, AWS KMS utilise deux méthodes d'établissement clés différentes. Le premier est défini comme C (1, 2, ECC DH) dans la Recommandation pour les schémas d'établissement de clés par paires utilisant la cryptographie à logarithme discret. Ce système dispose d'un initiateur avec une clé de signature statique. L'initiateur génère et signe une clé Diffie-Hellman (ECDH) à courbe elliptique éphémère, conçue pour un destinataire disposant d'une clé d'accord ECDH statique. Cette méthode utilise une seule clé éphémère et deux clés statiques utilisant ECDH. Ceci est la dérivation de l'étiquette C (1, 2, ECC DH). Cette méthode est parfois appelée « ECDH à passage unique ».

La deuxième méthode d'établissement de clé est C (2, 2, ECC, DH). Dans ce système, les deux parties disposent d'une clé de signature statique, et elles génèrent, signent et échangent une clé ECDH éphémère. Cette méthode utilise deux clés statiques et deux clés éphémères, chacune utilisant ECDH. Ceci est la dérivation de l'étiquette C (2, 2, ECC, DH). Cette méthode est parfois appelée « ECDH éphémère » ou « ECDHE ». Toutes les clés ECDH sont générées sur la courbe secp384r1 (NIST-P384).

Limite de sécurité des clés HSM

La limite de sécurité intérieure de AWS KMS est le HSM. La clé HSM est dotée d'une interface propriétaire et ne possède aucune autre interface physique active dans son état opérationnel. Une clé HSM opérationnelle est provisionnée lors de son initialisation avec les clés cryptographiques nécessaires à l'établissement de son rôle dans le domaine. Les éléments cryptographiques sensibles de la clé HSM sont uniquement stockés dans une mémoire volatile et effacés que lorsque la clé HSM quitte l'état opérationnel, notamment lors des arrêts ou réinitialisations prévus ou non.

Les opérations de l'API de la clé HSM sont authentifiées soit par des commandes individuelles, soit par une session confidentielle mutuellement authentifiée établie par un hôte de service.

Commandes signées en quorum

Les commandes signées par quorum sont émises par les opérateurs pour. HSMs Cette section décrit comment les commandes basées sur le quorum sont créées, signées et authentifiées. Ces règles sont assez simples. Par exemple, la commande Foo nécessite deux membres du rôle Bar pour être authentifiée. La création et la vérification d'une commande basée sur le quorum comporte trois

étapes. La première étape est la création initiale de la commande ; la seconde est la soumission à d'autres opérateurs pour signature ; et la troisième est la vérification et l'exécution.

Pour présenter les concepts, supposons qu'il existe un ensemble authentique de clés et de rôles publics {QOS_s} de l'opérateur, et un ensemble de règles de quorum QR = {$Command_i$, $Rule_{\{i, t\}}$} où chaque règle est un ensemble de rôles et le nombre minimum N {Role, N}. $_{t\ t}$ Afin qu'une commande respecte la règle de quorum, le jeu de données de la commande doit être signé par un ensemble d'opérateurs répertoriés dans {QOS_s} de façon à ce qu'ils répondent à l'une des règles répertoriées pour cette commande. Comme mentionné précédemment, l'ensemble des règles et des opérateurs du quorum sont stockés dans l'état du domaine et dans le jeton de domaine exporté.

Dans la pratique, un signataire initial signe la commande Sig_1 = Sign(dO_{p1}, Command). Un second opérateur signe également la commande Sig_2 = Sign(dO_{p2}, Command). Le message doublement signé est envoyé à une clé HSM pour exécution. La clé HSM effectue les tâches suivantes :

1. Pour chaque signature, il extrait la clé publique du signataire de l'état du domaine et vérifie la signature sur la commande.
2. Elle vérifie que l'ensemble des signataires satisfait à une règle pour la commande.

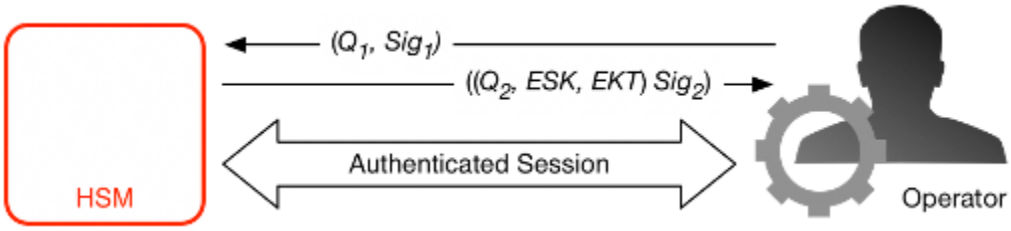
Sessions authentifiées

Vos principales opérations s'exécutent entre les AWS KMS hôtes orientés vers l'extérieur et le HSMs. Ces commandes concernent la création et l'utilisation de clés cryptographiques et la génération sécurisée de nombres aléatoire. Les commandes s'exécutent sur un canal authentifié par session entre les hôtes du service et le. HSMs Outre le besoin d'authenticité, ces sessions exigent la confidentialité. Les commandes exécutées sur ces sessions comprennent le retour de clés de données en texte clair et de messages déchiffrés qui vous sont destinés. Pour s'assurer que ces sessions ne peuvent pas être subverties par man-in-the-middle des attaques, les sessions sont authentifiées.

Ce protocole exécute un accord de clé ECDHE mutuellement authentifié entre la clé HSM et l'hôte de service. L'échange est initié par l'hôte de service et achevé par la clé HSM. La clé HSM renvoie également une clé de session (SK) chiffrée par la clé négociée et un jeton de clé exporté contenant la clé de session. Le jeton de clé exporté comporte une période de validité, après laquelle l'hôte de service devra renégocier une clé de session.

Un hôte de service est membre du domaine et possède une paire de clés de signature d'identité ($DHos_i$, $QHOS_i$) et une copie authentique des « clés publiques d'identité HSMs ». Il utilise son

ensemble de clés identité-signature pour négocier en toute sécurité une clé de session qui peut être utilisée entre l'hôte de service et toute clé HSM du domaine. Les jetons de clé exportés ont une période de validité qui leur est associée, après laquelle une nouvelle clé devra être négociée.



Le processus commence par la reconnaissance de l'hôte de service qui a besoin d'une clé de session pour envoyer et recevoir des flux de communications sensibles entre lui-même et un membre de clé HSM du domaine.

1. Un hôte de service génère une paire de clés éphémères ECDH $(d)_1$, $Q._1$) et la signe avec sa clé d'identité Sig1 = Sig(dOS, Q_1).
2. La clé HSM vérifie la signature sur la clé publique reçue à l'aide de son jeton de domaine actuel et crée une paire de clés éphémères ECDH $(d)_2$, $Q._2$). Il complète ensuite la recommandation pour ECDH-key-exchange les schémas d'établissement de clés par paires utilisant la cryptographie à logarithme discret pour former une clé AES-GCM 256 bits négociée. La clé HSM génère une nouvelle clé de session AES-GCM 256 bits. Elle chiffre la clé de session à l'aide de la clé négociée afin de constituer la clé de session chiffrée (ESK). Elle chiffre également la clé de session sous la clé de domaine en tant que jeton de clé exporté EKT. Enfin, elle signe une valeur de retour à l'aide de sa paire de clés d'identité Sig_2 = Sign(dHSK, (Q_2, ESK, EKT)).
3. L'hôte de service vérifie la signature sur les clés reçues à l'aide de son jeton de domaine actuel. L'hôte du service effectue ensuite l'échange de clés ECDH conformément à la recommandation pour les schémas d'établissement de clés par paires utilisant la cryptographie à logarithme discret. Il déchiffre ensuite la clé ESK afin d'obtenir la clé de session SK.

Au cours de la période de validité dans l'EKT, l'hôte de service peut utiliser la clé de session négociée SK pour envoyer des commandes chiffrées par enveloppe à la clé HSM. Chaque service-host-initiated commande de cette session authentifiée inclut l'EKT. La clé HSM répond en utilisant la même clé de session SK négociée.

Processus de réplication pour clés multi-régions

AWS KMS utilise un mécanisme de réplication entre régions pour copier le contenu clé d'une clé KMS d'un HSM d'un HSM d'un autre Région AWS vers un HSM d'un autre. Région AWS Pour que ce mécanisme fonctionne, la clé KMS qui est répliquée doit être une clé multi-Régions. Lors de la réplication d'une clé KMS d'une région à l'autre, les HSMs régions ne peuvent pas communiquer directement, car elles se trouvent dans des réseaux isolés. Au lieu de cela, les messages échangés pendant la réplication entre régions sont délivrés par un service proxy.

Lors de la réplication entre régions, chaque message généré par un AWS KMS HSM est signé cryptographiquement à l'aide d'une clé de signature de réplication. Les clés de signature de réplication (RSKs) sont des clés ECDSA sur la courbe NIST P-384. Chaque région possède au moins une RSK, et la composante publique de chaque RSK est partagée avec toutes les autres régions de la même AWS partition.

Le processus de réplication entre régions pour copier les éléments de clé de la région A à la région B fonctionne comme suit :

1. Le HSM de la région B génère une clé ECDH éphémère sur la courbe NIST P-384, Replication Agreement Key B (RAKB). La composante publique de RAKB est envoyée à un HSM de la région A par le service proxy.
2. Le HSM de la région A reçoit la composante publique de RAKB, puis génère une autre clé ECDH éphémère sur la courbe NIST P-384, Replication Agreement Key A (RAKA). Le HSM exécute le schéma d'établissement de clé ECDH sur RAKA et la composante publique de RAKB, et dérive une clé symétrique de la sortie, la Replication Wrapping Key (RWK). La clé RWK est utilisée pour chiffrer les éléments de clé de la clé KMS multi-régions en cours de réplication.
3. La composante publique de RAKA et les éléments de clé chiffrés avec la clé RWK sont envoyés au HSM de la région B via le service proxy.
4. Le HSM de la région B reçoit la composante publique de RAKA et les éléments de clé chiffrés à l'aide de la clé RWK. Le HSM dérive la clé RWK en exécutant le schéma d'établissement de clé ECDH sur RAKB et la composante publique de RAKA.
5. Le HSM de la région B utilise la clé RWK pour déchiffrer la clé de la région A.

Protection de la durabilité

La durabilité du service pour les clés générées par le service est assurée par l'utilisation du stockage non volatile multiple hors ligne HSMs des jetons de domaine exportés et du stockage redondant

des clés KMS cryptées. Les personnes hors ligne HSMs sont membres des domaines existants. À l'exception du fait de ne pas être en ligne et de participer aux opérations régulières du domaine, les personnes hors ligne HSMs apparaissent de la même manière dans l'état du domaine que les membres HSM existants.

La conception durable vise à protéger toutes les clés KMS d'une région en cas de perte à grande échelle des clés KMS en ligne HSMs ou de l'ensemble des clés KMS stockées dans notre système de stockage principal. AWS AWS KMS keys avec du matériel de clé importé ne sont pas inclus dans les protections de durabilité accordées aux autres clés KMS. En cas de panne à l'échelle de la région AWS KMS, le matériel clé importé devra peut-être être réimporté dans une clé KMS.

Les données hors ligne HSMs et les informations d'identification permettant d'y accéder sont stockées dans des coffres-forts situés dans des salles sécurisées surveillées situées dans plusieurs emplacements géographiques indépendants. Chaque coffre-fort nécessite au moins un agent AWS de sécurité et un AWS KMS opérateur, issus de deux équipes indépendantes AWS, pour obtenir ces matériaux. L'utilisation de ces matériaux est régie par une politique interne exigeant la présence d'un quorum d' AWS KMS opérateurs.

Historique du document

Cette rubrique décrit les mises à jour importantes apportées au Guide du développeur AWS Key Management Service .

Rubriques

Mises à jour récentes

Le tableau suivant décrit les modifications importantes apportées à cette documentation depuis janvier 2018. En plus des principales modifications répertoriées ici, nous mettons fréquemment à jour la documentation pour améliorer les descriptions et les exemples, et pour répondre aux commentaires que vous nous envoyez. Pour recevoir une notification concernant des modifications importantes, abonnez-vous au flux RSS.

Il peut être nécessaire de faire défiler horizontalement ou verticalement pour afficher toutes les données de ce tableau.

Modification	Description	Date
Support à double pile	AWS KMS prend en charge le double stack.	18 juin 2025
Mise à jour de la fonctionnalité	Prend en charge les signatures cryptographiques post-quantiques basées sur l'algorithme de signature numérique Module-Lattice (ML-DSA).	13 juin 2025
Rotation des clés importée	Vous pouvez effectuer une rotation à la demande des clés KMS à chiffrement symétrique avec le matériel clé importé (origine). `EXTERNAL`	5 juin 2025

Mise à jour de la fonctionnalité	Ajout de la prise en charge des clés KMS multirégionales dans les régions de Chine.	21 novembre 2024
AWS mise à jour des politiques gérée	Le rôle AWSKeyManagementServiceMultiRegionKeysServiceRolePolicylié au service a été mis à jour en ajoutant un ID d'instruction (`Sid`) à la politique gérée avec la version v2 de la politique.	21 novembre 2024
Changement de quota	Le taux de demandes par défaut a été augmenté pour ImportKeyMaterialet pour les DeleteImportedKeyMaterialdemandes.	23 juillet 2024
Changement de quota	Augmentation du taux de demandes d'opérations cryptographiques par défaut pour les clés KMS de chiffrement symétriques, les clés RSA KMS et les clés ECC et KMS. SM2	8 juillet 2024
Nouvelle fonction	Ajout `KeyUsage` d'un nouveau type `KEY_AGREEMENT` pour les clés KMS à courbe elliptique (ECC) et (régions de SM2 Chine uniquement) recommandées par le NIST et ajout de la prise en charge de la dérivation de secrets partagés.	13 juin 2024

Mises à jour de la rotation des clés	Ajout de la prise en charge des périodes de rotation personnalisées pour les rotations automatiques des touches, des rotations de touches à la demande et de la visibilité sur les rotations de vos matériaux clés.	12 avril 2024
Mises à jour de la politique gérée	De nouvelles autorisations ont été ajoutées `AWSKeyManagementServiceCustomKeyStoresServiceRolePolicy` qui AWS KMS permettent de surveiller les modifications apportées au VPC qui contient votre AWS CloudHSM cluster afin de fournir des messages d'erreur clairs en cas de défaillance. AWS KMS	10 novembre 2023
Mise à jour de la fonctionnalité	Ajout de la prise en charge du paramètre d'API `DryRun`.	5 juillet 2023
Mise à jour de la fonctionnalité	Ajout de la prise en charge de l'importation de matériel clé pour tous les types de AWS KMS clés, à l'exception des magasins de clés personnalisés.	5 juin 2023
Mise à jour de la fonctionnalité	Mises à jour de AWS KMS APIs for Nitro Enclaves	10 mars 2023

Mise à jour de la fonctionnalité	L'algorithme `RSAES_PKCS1_V1_5` d'encapsulage est obsolète. AWS KMS mettra fin à tout support d'`RSAES_PKCS1_V1_5` ici le 1er octobre 2023 conformément aux directives de gestion des clés cryptographiques du National Institute of Standards and Technology (NIST). Nous vous recommandons de commencer à utiliser un algorithme d'encapsulage différent immédiatement.	28 février 2023
Mise à jour de la fonctionnalité	Ajout de la prise en charge des magasins de clés externes, une fonctionnalité qui vous permet de protéger vos AWS ressources à l'aide de AWS clés cryptographiques extérieures à.	29 novembre 2022
Changement de quota	Le quota de AWS KMS keys ressources a été augmenté à 100 000 clés KMS dans chaque compte et région.	8 juillet 2022
Mise à jour des fonctionnalités	Ajout de la prise en charge des clés HMAC KMS en plus Régions AWS	8 juillet 2022
Nouvelle rubrique	La AWS Key Management Service rubrique Résilience a été ajoutée au chapitre sur la sécurité du guide du AWS KMS développeur.	14 juin 2022

Nouvelle fonction	Ajout de la prise en charge des AWS KMS clés et des opérations d'API qui génèrent et vérifient les codes HMAC.	19 avril 2022
Modification de la documentation	Remplacez le terme clé principale client (CMK) par AWS KMS key et clé KMS.	30 août 2021
Nouvelle fonction	Ajout de la prise en charge des clés multi-région, un ensemble de clés KMS interopérables dans différentes régions qui ont le même ID de clé et les mêmes éléments de clé. Vous pouvez utiliser des clés multi-région pour chiffrer des données dans une région et déchiffrer des données dans une autre région.	8 juin 2021
Nouvelle fonction	Ajout de la prise en charge du contrôle d'accès basé sur l'attribut (ABAC). Vous pouvez utiliser des balises et des alias pour contrôler l'accès à votre AWS KMS keys.	17 décembre 2020
Nouvelle fonction	Ajout de la prise en charge pour les politiques de point de terminaison d'un VPC.	9 juillet 2020
Nouveau contenu	Explique les propriétés de sécurité de AWS KMS.	18 juin 2020

Nouvelle fonction	Ajout de la prise en charge des clés de données asymétriques AWS KMS keys et asymétriques.	25 novembre 2019
Fonction mise à jour	Vous pouvez consulter la politique clé de Clés gérées par AWS dans la AWS KMS console. Cette fonction était auparavant limitée aux clés gérées par le client.	15 novembre 2019
Nouvelle fonction	Explique comment utiliser des algorithmes échange de clés post-quantiques hybrides dans TLS pour vos appels vers AWS KMS.	4 novembre 2019
Changement de quota	Les quotas de ressources ont été augmentés pour certaines APIs entités qui gèrent les clés KMS.	18 septembre 2019
Changement de quota	Modification des quotas de ressources pour les clés KMS, les alias et les octrois par clé KMS.	27 mars 2019
Changement de quota	Modification du quota de demande par seconde partagé pour les opération s de chiffrement qui utilisent les AWS KMS keys dans un magasin de clés personnalisé.	7 mars 2019

Nouvelle fonction	Explique comment créer et gérer des magasins de clés AWS KMS personnalisés. Chaque magasin de clés est soutenu par un AWS CloudHSM cluster que vous possédez et contrôlez.	26 novembre 2018
Nouvelle console	Explique comment utiliser la nouvelle AWS KMS console, qui est indépendante de la console IAM. La console d'origine, et les instructi ons pour l'utiliser, resteront disponibles pendant une brève période pour vous donner le temps de vous familiariser avec la nouvelle console.	7 novembre 2018
Changement de quota	Modification du quota de demandes partagées pour l'utilisation de AWS KMS keys.	21 août 2018
Nouveau contenu	Explique AWS Secrets Manager comment utiliser AWS KMS les clés pour chiffrer la valeur secrète d'un secret.	13 juillet 2018
Nouveau contenu	Explique comment DynamoDB prend en charge son AWS KMS AWS KMS keys option de chiffrement côté serveur.	23 mai 2018

Nouvelle fonction	Explique comment utiliser un point de terminaison privé dans votre VPC pour vous y connecter directement AWS KMS, au lieu de vous connecter via Internet.	22 janvier 2018

Mises à jour antérieures

Le tableau suivant décrit les modifications importantes apportées au Guide du AWS Key Management Service développeur avant 2018.

Il peut être nécessaire de faire défiler horizontalement ou verticalement pour afficher toutes les données de ce tableau.

Modification	Description	Date
Nouveau contenu	Ajout de la documentation sur Tags dans AWS KMS.	15 février 2017
Nouveau contenu	Ajout de la documentation sur Moniteur AWS KMS keys et Surveillez les clés KMS avec Amazon CloudWatch.	31 août 2016
Nouveau contenu	Ajout de la documentation sur Éléments de clé importés.	11 août 2016
Nouveau contenu	Ajout de la documentation suivante : Politiques IAM, Référence des autorisations et Clés de condition.	5 juillet 2016
Mettre à jour	Mise à jour de certaines parties de la documentation dans le chapitre Accès aux clés KMS et autorisations.	5 juillet 2016

Modification	Description	Date
Mettre à jour	Mise à jour de la page Quotas pour refléter les nouveaux quotas par défaut.	31 mai 2016
Mettre à jour	Mise à jour de la page Quotas pour refléter les nouveaux quotas par défaut et mise à jour de la documentation sur le jeton d'octroi pour améliorer la clarté et la précision.	11 avril 2016
Nouveau contenu	Ajout de la documentation sur Attribution à plusieurs principaux IAM de l'autorisation d'accès à une clé KMS et Utilisation de la condition d'adresse IP.	17 février 2016
Mettre à jour	Mise à jour des pages Politiques clés en AWS KMS et Modifier une politique clé pour améliorer la clarté et la précision.	17 février 2016
Mettre à jour	Mise à jour des pages thématiques sur la gestion des clés KMS pour améliorer la clarté.	5 janvier 2016
Nouveau contenu	Ajout de documentation sur CloudTrail.	18 novembre 2015
Nouveau contenu	Ajout d'instructions pour Modifier une politique clé.	18 novembre 2015

Modification	Description	Date
Mettre à jour	Mise à jour de la documentation sur l'utilisation d'Amazon Relational Database Service. AWS KMS	18 novembre 2015
Nouveau contenu	Ajout de documentation sur Amazon WorkSpaces.	6 novembre 2015
Mettre à jour	Mise à jour de la page Politiques clés en AWS KMS pour améliorer la clarté.	22 octobre 2015
Nouveau contenu	Ajout de la documentation sur Supprimer un AWS KMS key, y compris de la documentation de prise en charge sur Créer une alarme et Déterminer l'utilisation passée d'une clé KMS.	15 octobre 2015
Nouveau contenu	Ajout de la documentation sur Déterminer l'accès à AWS KMS keys.	15 octobre 2015
Nouveau contenu	Ajout de la documentation sur États clés des AWS KMS clés.	15 octobre 2015
Nouveau contenu	Ajout de documentation sur Amazon Simple Email Service.	1er octobre 2015
Mettre à jour	Mise à jour de la page Quotas pour expliquer les nouveaux quotas de demande.	31 août 2015

Modification	Description	Date
Nouveau contenu	Ajout d'informations sur les frais d'utilisation AWS KMS. Veuillez consulter Tarification AWS KMS.	14 août 2015
Nouveau contenu	Des quotas de demandes ont été ajoutés au AWS KMS Quotas.	11 juin 2015
Nouveau contenu	Ajout d'un nouvel exemple de code Java illustrant l'utilisation de l'opération `UpdateAlias` .	1er juin 2015
Mettre à jour	Déplacement de la table des régions AWS Key Management Service vers la Références générales AWS.	29 mai 2015
Nouveau contenu	Ajout de la documentation sur Comment Amazon EMR utilise AWS KMS.	28 janvier 2015
Nouveau contenu	Ajout de documentation sur Amazon WorkMail.	28 janvier 2015
Nouveau contenu	Ajout de documentation sur le mode d'utilisation d'Amazon Relational Database Service. AWS KMS	6 janvier 2015
Nouveau contenu	Ajout de documentation sur Amazon Elastic Transcoder.	24 novembre 2014
Nouveau guide	Présentation du Guide du développeur AWS Key Management Service .	12 novembre 2014

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.