Guide de l’utilisateur

AWS Cryptographie des paiements

AWS Cryptographie des paiements: Guide de l’utilisateur

Table of Contents

Qu'est-ce que la cryptographie des AWS paiements ?

AWS La cryptographie des paiements est un AWS service géré qui donne accès aux fonctions cryptographiques et à la gestion des clés utilisées dans le traitement des paiements conformément aux normes du secteur des cartes de paiement (PCI) sans que vous ayez à vous procurer des instances HSM de paiement dédiées. AWS La cryptographie des paiements permet aux clients exécutant des fonctions de paiement telles que les acquéreurs, les facilitateurs de paiement, les réseaux, les commutateurs, les processeurs et les banques de rapprocher leurs opérations cryptographiques de paiement des applications du cloud et de minimiser leur dépendance à l'égard des centres de données auxiliaires ou des installations de colocation proposant des paiements dédiés. HSMs

Le service est conçu pour répondre aux règles applicables du secteur, notamment les normes PCI PIN, PCI P2PE et PCI DSS, et le service utilise du matériel certifié PCI PTS HSM V3 et FIPS 140-2 niveau 3. Il est conçu pour supporter une faible latence ainsi que des niveaux élevés de disponibilité et de résilience. AWS La cryptographie des paiements est totalement élastique et élimine de nombreuses exigences opérationnelles sur site HSMs, telles que la nécessité de fournir du matériel, de gérer en toute sécurité le matériel clé et de maintenir des sauvegardes d'urgence dans des installations sécurisées. AWS La cryptographie des paiements vous offre également la possibilité de partager des clés avec vos partenaires par voie électronique, éliminant ainsi le besoin de partager des composants de texte en clair sur papier.

Vous pouvez utiliser l'API AWS Payment Cryptography Control Plane pour créer et gérer des clés.

Vous pouvez utiliser l'API AWS Payment Cryptography Data Plane pour utiliser des clés de chiffrement pour le traitement des transactions liées au paiement et les opérations cryptographiques associées.

AWS La cryptographie des paiements fournit des fonctionnalités importantes que vous pouvez utiliser pour gérer vos clés :

- Créez et gérez des clés de cryptographie de AWS paiement symétriques et asymétriques, notamment des clés TDES, AES et RSA, et spécifiez leur objectif, par exemple pour la génération de CVV ou la dérivation de clés DUKPT.
- Stockez automatiquement vos clés AWS de cryptographie de paiement en toute sécurité, protégées par des modules de sécurité matériels (HSMs) tout en garantissant la séparation des clés entre les cas d'utilisation.

- Créez, supprimez, listez et mettez à jour des alias, qui sont des « noms conviviaux » qui peuvent être utilisés pour accéder ou contrôler l'accès à vos clés de cryptographie AWS de paiement.
- Marquez vos clés AWS de cryptographie de paiement pour l'identification, le regroupement, l'automatisation, le contrôle d'accès et le suivi des coûts.
- Importez et exportez des clés symétriques entre AWS Payment Cryptography et votre HSM (ou des tiers) à l'aide de clés de chiffrement (KEK) conformément à la norme TR-31 (Interoperable Secure Key Exchange Key Block Specification).
- Importez et exportez des clés de chiffrement à clé symétrique (KEK) entre la cryptographie des AWS paiements et d'autres systèmes à l'aide de paires de clés asymétriques, puis utilisez des moyens électroniques tels que le TR-34 (méthode de distribution de clés symétriques à l'aide de techniques asymétriques).

Vous pouvez utiliser vos clés de chiffrement des AWS paiements dans le cadre d'opérations cryptographiques, telles que :

- Chiffrez, déchiffrez et rechiffrez les données à l'aide de clés de cryptographie de paiement symétriques ou asymétriques. AWS
- Traduisez en toute sécurité les données sensibles (telles que les codes PIN du titulaire de la carte) entre les clés de chiffrement sans exposer le texte clair conformément aux règles PCI PIN.
- Générez ou validez les données du titulaire de la carte, telles que CVV CVV2 ou ARQC.
- Générez et validez les codes PIN du titulaire de la carte.
- Générez ou validez des signatures MAC.

Concepts

Découvrez les termes et concepts de base utilisés dans le AWS domaine de la cryptographie des paiements et découvrez comment vous pouvez les utiliser pour protéger vos données.

Alias

> Nom convivial associé à une clé de chiffrement des AWS paiements. L'alias peut être utilisé de manière interchangeable avec l'ARN de la clé dans de nombreuses opérations de l'API de cryptographie des AWS paiements. Les alias permettent de faire pivoter ou de modifier les clés sans affecter le code de votre application. Le nom d'alias est une chaîne comportant jusqu'à 256 caractères. Il identifie de manière unique une clé de cryptographie AWS de paiement

associée au sein d'un compte et d'une région. Dans AWS Payment Cryptography, les noms d'alias commencent `alias/` toujours par.

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

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

Par exemple :

```
alias/sampleAlias2
```

ARN de clé

La clé ARN est le nom de ressource Amazon (ARN) d'une entrée clé dans AWS Payment Cryptography. Il s'agit d'un identifiant unique et complet pour la clé de cryptographie des AWS paiements. Un ARN clé comprend une Compte AWS région et un identifiant généré de manière aléatoire. L'ARN n'est pas lié ou dérivé du matériau clé. Comme elles sont automatiquement attribuées lors des opérations de création ou d'importation, ces valeurs ne sont pas idempotentes. Si vous importez plusieurs fois la même clé, plusieurs clés ARNs auront leur propre cycle de vie.

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

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

Voici un exemple d'ARN de clé :

```
arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h
```

Identifiant clé

Un identifiant de clé est une référence à une clé et l'une (ou plusieurs) d'entre elles sont des entrées typiques des opérations de cryptographie des AWS paiements. Les identifiants de clé valides peuvent être un ARN de clé ou un alias de clé.

AWS Clés de chiffrement des paiements

AWS Les clés de cryptographie (clés) de paiement sont utilisées pour toutes les fonctions cryptographiques. Les clés sont soit générées directement par vous à l'aide du raccourci clavier Create, soit ajoutées au système en appelant key import. L'origine d'une clé peut être déterminée

en examinant l'attribut KeyOrigin. AWS La cryptographie des paiements prend également en charge les clés dérivées ou intermédiaires utilisées lors d'opérations cryptographiques telles que celles utilisées par DUKPT.

Ces clés ont des attributs immuables et mutables définis lors de leur création. Les attributs, tels que l'algorithme, la longueur et l'utilisation, sont définis lors de la création et ne peuvent pas être modifiés. D'autres, comme la date d'entrée en vigueur ou la date d'expiration, peuvent être modifiées. Consultez la référence de l'API de cryptographie des AWS paiements pour obtenir la liste complète des attributs des clés de cryptographie des AWS paiements.

AWS Les clés de cryptographie de paiement ont des types de clés, principalement définis par la norme ANSI X9 TR 31, qui limitent leur utilisation aux fins prévues, conformément à l'exigence 19 du PCI PIN v3.1.

Les attributs sont liés aux clés à l'aide de blocs de clés lorsqu'ils sont stockés, partagés avec d'autres comptes ou exportés conformément à l'exigence 18-3 du PCI PIN v3.1.

Les clés sont identifiées sur la plateforme AWS de cryptographie des paiements à l'aide d'une valeur unique appelée clé Amazon Resource Name (ARN).

> **Note**
>
> ARNLa clé est générée lorsqu'une clé est initialement créée ou importée dans le service de cryptographie des AWS paiements. Ainsi, si vous ajoutez le même élément clé plusieurs fois à l'aide de la fonctionnalité de clé d'importation, le même matériau clé sera situé sous plusieurs clés, ARNS mais chacune aura un cycle de vie de clé différent.

Terminologie de l'industrie

Rubriques

Types de clés courants

AWS Clé de cryptographie de paiement

Une clé cryptographique de AWS paiement existe en un seul Région AWS. Il comprend des métadonnées clés et du matériel stockés dans le service de cryptographie des AWS paiements. Une clé peut être importée depuis une source externe sous forme de bloc TR-31 clé ou générée par le service de cryptographie des AWS paiements.

POINÇON

Une clé de travail de l'acquéreur (AWK) est une clé généralement utilisée pour échanger des données entre un acquirer/acquirer processeur et un réseau (tel que Visa ou Mastercard). Historiquement, AWK utilise 3DES pour le chiffrement et était représenté sous la forme TR31_P0_PIN_ENCRYPTION_KEY.

BDK

Une clé de dérivation de base (BDK) est une clé de travail utilisée pour dériver les clés suivantes. Elle est couramment utilisée dans le cadre des processus PCI PIN et PCI P2PE DUKPT. Il est noté TR31_B0_BASE_DERIVATION_KEY.

CLÉ CMK

Une clé principale de carte (CMK) est une ou plusieurs clés spécifiques à une carte généralement dérivées d'une clé principale d'émetteur, d'un PAN et d'un PSN et sont généralement des clés 3DES. Ces clés sont stockées sur la puce EMV lors de la personnalisation. Les clés AC, SMI et SMC sont des exemples de CMK.

CMK-AC

Une clé de cryptogramme d'application (AC) est utilisée dans le cadre des transactions EMV pour générer le cryptogramme de transaction et constitue un type de clé principale de carte.

CMK-SMI

Une clé d'intégrité de la messagerie sécurisée (SMI) est utilisée dans le cadre de l'EMV pour vérifier l'intégrité des charges utiles envoyées à la carte via MAC, telles que les scripts de mise à jour du code PIN. Il s'agit d'un type de clé principale de carte.

CMK-SMC

Une clé de confidentialité des messages sécurisés (SMC) est utilisée dans le cadre de l'EMV pour chiffrer les données envoyées à la carte, telles que les mises à jour du code PIN. Il s'agit d'un type de clé principale de carte.

CVK

Une clé de vérification de carte (CVK) est une clé utilisée pour générer des valeurs CVV, CVV2 et similaires à l'aide d'un algorithme défini, ainsi que pour valider une entrée. Il est désigné sous le nom de TR31_C0_CARD_VERIFICATION_KEY.

IMK

Une clé principale de l'émetteur (IMK) est une clé principale utilisée dans le cadre de la personnalisation des cartes à puce EMV. Il y aura généralement 3 IMK, une pour chacune des clés AC (cryptogramme), SMI (clé principale de script pour integrity/signature) et SMC (clé principale de script pour). confidentiality/encryption

cinématique inverse

Une clé initiale (IK) est la première clé utilisée dans le processus DUKPT et dérive de la clé de dérivation de base (BDK). Aucune transaction n'est traitée sur cette clé, mais elle est utilisée pour dériver les futures clés qui seront utilisées pour les transactions. La méthode de dérivation pour créer un IK a été définie dans X9.24-1:2017. Lorsqu'un TDES BDK est utilisé, c'est la norme applicable et l'IK X9.24-1:2009 est remplacé par la clé de cryptage IPEK (Initial Pin Encryption Key).

IPEK

Une clé de chiffrement par code PIN initial (IPEK) est la clé initiale utilisée dans le processus DUKPT et dérive de la clé de dérivation de base (BDK). Aucune transaction n'est traitée sur cette clé, mais elle est utilisée pour dériver les futures clés qui seront utilisées pour les transactions. IPEK est un terme impropre car cette clé peut également être utilisée pour dériver le cryptage des données et les clés Mac. La méthode de dérivation pour créer un IPEK a été définie dans. X9.24-1:2009 Lorsqu'un AES BDK est utilisé, c' X9.24-1:2017 est la norme applicable et l'IPEK est remplacé par la clé initiale (IK).

IWK

Une clé de travail de l'émetteur (IWK) est une clé généralement utilisée pour échanger des données entre un issuer/issuer processeur et un réseau (tel que Visa ou Mastercard).

Historiquement, IWK utilise 3DES pour le chiffrement et est représenté sous la forme TR31_P0_PIN_ENCRYPTION_KEY.

KBPK

Une clé de chiffrement par blocs de clés (KBPK) est un type de clé symétrique utilisée pour protéger les blocs de clés et donc wrap/encrypt les autres clés. Un KBPK est similaire à un KEK, mais un KEK protège directement le contenu clé alors que dans TR-31 des schémas similaires, le KBPK ne protège qu'indirectement la clé de travail. Lors de l'utilisation TR-31, TR31_K1_KEY_BLOCK_PROTECTION_KEY est le type de clé approprié, bien que TR31_K0_KEY_ENCRYPTION_KEY soit compatible de manière interchangeable à des fins historiques.

KEK

Une clé de chiffrement (KEK) est une clé utilisée pour chiffrer d'autres clés à des fins de transmission ou de stockage. Les clés destinées à protéger d'autres clés ont généralement la valeur KeyUsage TR31_K0_KEY_ENCRYPTION_KEY conformément à la norme. TR-31

PEK

Une clé de cryptage PIN (PEK) est un type de clé fonctionnelle utilisée pour chiffrer des codes PIN à des fins de stockage ou de transmission entre deux parties. IWK et AWK sont deux exemples d'utilisations spécifiques des clés de chiffrement par code PIN. Ces clés sont représentées sous la forme TR31_P0_PIN_ENCRYPTION_KEY.

PGK

PGK (Pin Generation Key) est un autre nom pour une clé de vérification par code PIN. Il n'est pas réellement utilisé pour générer des épingles (qui sont par défaut des nombres cryptographiquement aléatoires) mais plutôt pour générer des valeurs de vérification telles que le PVV.

PRK

La clé de région principale est la source de réplication officielle pour une clé de cryptographie de paiement donnée pour laquelle la réplication a été activée. PRK est une référence à un rôle clé de cryptographie des paiements source dans une configuration de réplication de Multi-Region clés. Lorsque la réplication est activée sur une clé de cryptographie de paiement, elle est appelée PRK pour cette configuration de réplication de clés spécifique.

PVK

Une clé de vérification du code PIN (PVK) est un type de clé fonctionnelle utilisée pour générer des valeurs de vérification du code PIN telles que le code PVV. Les deux types les plus courants sont le TR31_V1_IBM3624_PIN_VERIFICATION_KEY utilisé pour générer les valeurs de décalage IBM3624 et le TR31_V2_VISA_PIN_VERIFICATION_KEY utilisé pour les valeurs de vérification. Visa/ABA Cela peut également être connu sous le nom de clé de génération de code PIN.

RRK

Les clés de région de réplication sont le matériel clé répliqué et les métadonnées copiées de manière sécurisée depuis le PRK vers une réplique configurée. Région AWS Un RRK est une réplique en lecture seule d'une clé de cryptographie de paiement. RRK est une référence au rôle joué par une clé spécifique dans une configuration de réplication de Multi-Region clés. Toute modification importante des métadonnées, y compris les paramètres de réplication, doit être appliquée au PRK.

Autres termes

ARQC

Le cryptogramme de demande d'autorisation (ARQC) est un cryptogramme généré au moment de la transaction par une carte à puce standard EMV (ou une implémentation sans contact équivalente). Généralement, un ARQC est généré par une carte à puce et transmis à un émetteur ou à son agent pour vérification au moment de la transaction.

CVV

Une valeur de vérification de carte est une valeur secrète statique qui était traditionnellement intégrée sur une bande magnétique et utilisée pour valider l'authenticité d'une transaction. L'algorithme est également utilisé à d'autres fins telles que iCVV, CAVV, CVV2. Il se peut qu'il ne soit pas intégré de cette manière pour d'autres cas d'utilisation.

CVV2

Une valeur de vérification de carte 2 est une valeur secrète statique qui était traditionnellement imprimée au recto (ou au verso) d'une carte de paiement et utilisée pour vérifier l'authenticité des paiements non présentés par carte (par exemple au téléphone ou en ligne). Il utilise le même algorithme que le CVV mais le code de service est défini sur 000.

iCVV

iCVV est une CVV2-like valeur mais incorporée aux données équivalentes à track2 sur une carte EMV (Chip). Cette valeur est calculée à l'aide d'un code de service 999 et est différente de celle utilisée CVV1/CVV2 pour empêcher que des informations volées ne soient utilisées pour créer de nouveaux identifiants de paiement d'un type différent. Par exemple, si des données de transaction par puce ont été obtenues, il n'est pas possible de les utiliser pour générer une bande magnétique (CVV1) ou pour des achats en ligne (CVV2).

Il utilise une ??? clé

DUKPT

La clé unique dérivée par transaction (DUKPT) est une norme de gestion des clés généralement utilisée pour définir l'utilisation de clés de chiffrement à usage unique sur un support physique. POS/POI Historiquement, DUKPT utilise 3DES pour le chiffrement. La norme industrielle pour le DUKPT est définie dans l'ANSI. X9.24-3-2017

ECC

L'ECC (Elliptic Curve Cryptography) est un système de cryptographie à clé publique qui utilise les mathématiques des courbes elliptiques pour créer des clés de chiffrement. L'ECC fournit le même niveau de sécurité que les méthodes traditionnelles telles que le RSA, mais avec des clés beaucoup plus courtes, offrant ainsi une sécurité équivalente de manière plus efficace. Cela est particulièrement pertinent pour les cas d'utilisation où le RSA n'est pas une solution pratique (longueur de clé RSA > 4096 bits). AWS La cryptographie des paiements prend en charge les courbes définies par le NIST pour être utilisées dans les opérations ECDH.

ECDH

L'ECDH (Elliptic Curve Diffie-Hellman) est un protocole d'accord clé qui permet à deux parties d'établir un secret partagé (tel qu'un KEK ou un PEK). Dans l'ECDH, les parties A et B possèdent chacune leurs propres paires de clés publique-privée et échangent des clés publiques (sous forme de certificats pour la cryptographie des AWS paiements) ainsi que des métadonnées de dérivation des clés (méthode de dérivation, type de hachage et informations partagées). Les deux parties multiplient leur clé privée par la clé publique de l'autre et grâce aux propriétés de la courbe elliptique, les deux parties sont en mesure de dériver (générer) la clé obtenue.

EMV

EMV (à l'origine Europay, Mastercard, Visa) est un organisme technique qui travaille avec les parties prenantes du secteur des paiements pour créer des normes et des technologies de

paiement interopérables. Un exemple de norme concerne les chip/contactless cartes et les terminaux de paiement avec lesquels elles interagissent, y compris la cryptographie utilisée. La dérivation de clés EMV fait référence à des méthodes permettant de générer des clés uniques pour chaque carte de paiement sur la base d'un ensemble initial de clés tel qu'un `IMK`

HSM

Un module de sécurité matérielle (HSM) est un dispositif physique qui protège les opérations cryptographiques (par exemple, le chiffrement, le déchiffrement et les signatures numériques) ainsi que les clés sous-jacentes utilisées pour ces opérations.

KCAAS

Un dépositaire de clés en tant que service (KCAAS) fournit une variété de services liés à la gestion des clés. Pour les clés de paiement, ils peuvent généralement convertir des composants clés sur papier en formulaires électroniques compatibles avec la cryptographie des AWS paiements ou convertir des clés protégées électroniquement en composants papier qui peuvent être requis par certains fournisseurs. Ils peuvent également fournir des services de dépôt fiduciaire pour les clés dont la perte serait préjudiciable à vos opérations en cours. Les fournisseurs de KCAAS sont en mesure d'aider les clients à se décharger de la charge opérationnelle liée à la gestion des informations clés en dehors d'un service sécurisé tel que la cryptographie des AWS paiements, conformément aux normes PCI DSS, PCI PIN et PCI P2PE. AWS La cryptographie des paiements offre Échange de clés physiques une fonctionnalité KCAAS intégrée pour convertir des composants clés sur papier en format électronique.

KCV

La valeur de contrôle des clés (KCV) fait référence à une variété de méthodes de somme de contrôle principalement utilisées pour comparer les clés les unes aux autres sans avoir accès au contenu de la clé elle-même. Les KCV ont également été utilisés pour la validation de l'intégrité (en particulier lors de l'échange de clés), bien que ce rôle soit désormais inclus dans les formats de blocs de clés tels que `TR-31`. Pour les clés TDES, le KCV est calculé en chiffrant 8 octets, chacun ayant une valeur de zéro, avec la clé à vérifier et en conservant les 3 octets d'ordre le plus élevé du résultat chiffré. Pour les clés AES, le KCV est calculé à l'aide d'un algorithme CMAC où les données d'entrée sont de 16 octets de zéro et en conservant les 3 octets d'ordre le plus élevé du résultat chiffré.

KDH

Un hôte de distribution de clés (KDH) est un appareil ou un système qui envoie des clés dans le cadre d'un processus d'échange de clés tel que TR-34. Lors de l'envoi de clés depuis AWS Payment Cryptography, il est considéré comme le KDH.

KIF

Une installation d'injection de clés (KIF) est une installation sécurisée utilisée pour initialiser les terminaux de paiement, notamment pour les charger avec des clés de chiffrement.

KRD

Un dispositif de réception de clés (KRD) est un appareil qui reçoit des clés dans le cadre d'un processus d'échange de clés tel que TR-34. Lorsque vous envoyez des clés à AWS Payment Cryptography, celle-ci est considérée comme le KRD.

KSN

Un numéro de série de clé (KSN) est une valeur utilisée comme entrée dans DUKPT pour créer des clés encryption/decryption de chiffrement uniques par transaction. Le KSN se compose généralement d'un identifiant BDK, d'un identifiant de terminal semi-unique ainsi que d'un compteur de transactions qui s'incrémente à chaque transition traitée sur un terminal de paiement donné. Par exemple X9.24, pour le TDES, le KSN à 10 octets comprend généralement 24 bits pour l'identifiant du jeu de clés, 19 bits pour l'identifiant du terminal et 21 bits pour le compteur de transactions, bien que la limite entre l'identifiant du jeu de clés et l'identifiant du terminal n'ait aucun impact sur le fonctionnement de la cryptographie des AWS paiements. Pour AES, le KSN à 12 octets comprend généralement 32 bits pour l'identifiant BDK, 32 bits pour l'identifiant de dérivation (ID) et 32 bits pour le compteur de transactions.

mPOC

Le mPOC (point de vente mobile sur matériel commercial) est une norme PCI qui répond aux exigences de sécurité des solutions permettant aux commerçants d'accepter les codes PIN des titulaires de carte ou les paiements sans contact à l'aide d'un smartphone ou d'autres appareils mobiles commerciaux prêts à l'emploi (COTS).

PAN

Un numéro de compte principal (PAN) est un identifiant unique pour un compte tel qu'une carte de crédit ou de débit. Généralement de 13 à 19 chiffres. Les 6 à 8 premiers chiffres identifient le réseau et la banque émettrice.

Bloc PIN

Bloc de données contenant un code PIN pendant le traitement ou la transmission ainsi que d'autres éléments de données. Les formats de bloc PIN normalisent le contenu du bloc PIN et la manière dont il peut être traité pour récupérer le code PIN. La plupart des blocs PIN sont

composés du code PIN, de la longueur du code PIN, et contiennent souvent une partie ou la totalité du PAN. AWS La cryptographie des paiements prend en charge les formats ISO 9564-1 0, 1, 3 et 4. Le format 4 est requis pour les clés AES. Lors de la vérification ou de la traduction des codes PIN, il est nécessaire de spécifier le bloc PIN des données entrantes ou sortantes.

POI

Le point d'interaction (POI), également fréquemment utilisé de manière anonyme avec le point de vente (POS), est le périphérique matériel avec lequel le titulaire de la carte interagit pour présenter son identifiant de paiement. Un exemple de POI est le terminal physique d'un commerçant. Pour consulter la liste des terminaux PCI PTS POI certifiés, consultez le site Web PCI.

PSN

Le numéro de séquence PAN (PSN) est une valeur numérique utilisée pour différencier plusieurs cartes émises avec le même PAN.

Clé publique

Lors de l'utilisation de chiffrements asymétriques (RSA, ECC), la clé publique est le composant public d'une paire de clés publique-privée. La clé publique peut être partagée et distribuée aux entités qui ont besoin de chiffrer les données pour le propriétaire de la paire de clés publique-privée. Pour les opérations de signature numérique, la clé publique est utilisée pour vérifier la signature.

Clé privée

Lors de l'utilisation de chiffrements asymétriques (RSA, ECC), la clé privée est le composant privé d'une paire de clés publique-privée. La clé privée est utilisée pour déchiffrer des données ou créer des signatures numériques. À l'instar des clés de cryptographie AWS de paiement symétriques, les clés privées sont créées de manière sécurisée par les HSM. Ils sont déchiffrés uniquement dans la mémoire volatile du HSM et uniquement pendant le temps nécessaire au traitement de votre demande cryptographique.

PVV

Une valeur de vérification du code PIN (PVV) est un type de sortie cryptographique qui peut être utilisé pour vérifier un code PIN sans enregistrer le code PIN réel. Bien qu'il s'agisse d'un terme générique, dans le contexte de la cryptographie des AWS paiements, PVV fait référence à la méthode PVV Visa ou ABA. Ce PVV est un numéro à quatre chiffres dont les entrées sont le numéro de carte, le numéro de séquence Pan, le PAN lui-même et une clé de vérification du code

PIN. Au cours de la phase de validation, AWS Payment Cryptography recrée en interne le PVV à l'aide des données de transaction et le compare à nouveau à la valeur enregistrée par le client de AWS Payment Cryptography. De cette façon, il est conceptuellement similaire à un hachage cryptographique ou à un MAC.

RSA Wrap/Unwrap

L'encapsulation RSA utilise une clé asymétrique pour encapsuler une clé symétrique (telle qu'une clé TDES) afin de la transmettre à un autre système. Seul le système possédant la clé privée correspondante peut déchiffrer la charge utile et charger la clé symétrique. À l'inverse, RSA unwrap déchiffre en toute sécurité une clé chiffrée à l'aide de RSA, puis la charge dans le chiffrement des paiements. AWS L'encapsulation RSA est une méthode d'échange de clés de bas niveau qui ne transmet pas les clés sous forme de blocs de clés et n'utilise pas de signature de charge utile par l'expéditeur. Des contrôles alternatifs doivent être envisagés pour vérifier si les attributs clés ne sont pas modifiés.

TR-34 utilise également RSA en interne, mais il s'agit d'un format distinct et n'est pas interopérable.

TR-31

TR-31 (officiellement défini comme ANSI X9 TR 31) est un format de bloc clé défini par l'American National Standards Institute (ANSI) pour permettre de définir les attributs clés dans la même structure de données que les données clés elles-mêmes. Le format du bloc TR-31 clé définit un ensemble d'attributs clés liés à la clé afin qu'ils soient maintenus ensemble. AWS La cryptographie des paiements utilise des termes TR-31 standardisés dans la mesure du possible pour garantir une séparation correcte des clés et un objectif clé. TR-31 a été remplacé par l'ANSI. X9.143-2022

TR-34

TR-34 est une implémentation de l'ANSI X9.24-2 qui décrit un protocole permettant de distribuer de manière sécurisée des clés symétriques (telles que 3DES et AES) à l'aide de techniques asymétriques (telles que RSA). AWS La cryptographie des paiements utilise TR-34 des méthodes permettant l'importation et l'exportation sécurisées de clés.

X9.143

X9.143 est un format de bloc clé défini par l'American National Standards Institute (ANSI) pour permettre de sécuriser une clé et des attributs clés dans la même structure de données. Le format du bloc clé définit un ensemble d'attributs clés liés à la clé afin qu'ils soient maintenus ensemble. AWS La cryptographie des paiements utilise des termes X9.143 standardisés dans la mesure

du possible pour garantir une séparation correcte des clés et un objectif clé. X9.143 remplace la TR-31proposition précédente bien que, dans la plupart des cas, ils soient rétrocompatibles et que les termes soient souvent utilisés de manière interchangeable.

Services connexes

AWS Key Management Service

AWS Le service de gestion des clés (AWS KMS) est un service géré qui vous permet de créer et de contrôler facilement les clés cryptographiques utilisées pour protéger vos données. AWS KMS utilise des modules de sécurité matériels (HSMs) pour protéger et valider vos clés AWS KMS.

AWS CloudHSM

AWS CloudHSM fournit aux clients des instances HSM dédiées à usage général dans le AWS cloud. AWS CloudHSM peut fournir diverses fonctions cryptographiques telles que la création de clés, la signature de données ou le chiffrement et le déchiffrement de données.

Pour plus d'informations

- Pour en savoir plus sur les termes et concepts utilisés dans la cryptographie des AWS paiements, consultez la section Concepts AWS de cryptographie des paiements.
- Pour plus d'informations sur l'API AWS Payment Cryptography Control Plane, reportez-vous à la section Référence de l'API AWS Payment Cryptography Control Plane.
- Pour plus d'informations sur l'API du plan de données de cryptographie des AWS paiements, consultez la section Référence de l'API du plan de données de cryptographie des AWS paiements.
- Pour obtenir des informations techniques détaillées sur la manière dont la cryptographie des AWS paiements utilise la cryptographie et sécurise les clés de cryptographie des AWS paiements, consultez la section Détails cryptographiques.

Points de terminaison pour AWS Payment Cryptography

Pour vous connecter par programmation AWS Payment Cryptography, vous utilisez un point de terminaison, l'URL du point d'entrée du service. Les AWS SDK et les outils de ligne de commande utilisent automatiquement le point de terminaison par défaut pour le service en Région AWS fonction du contexte régional d'une demande. Il n'est donc généralement pas nécessaire de définir

explicitement ces valeurs. Si nécessaire, vous pouvez spécifier un point de terminaison différent pour vos demandes d'API.

Points de terminaison du plan de contrôle

Nom de la région	Région	Point de terminaison	Protocole
US East (Ohio)	us-east-2	controlplane.payment-cryptography.us-east-2.amazonaws.com controlplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS
USA Est (Virginie du Nord)	us-east-1	controlplane.payment-cryptography.us-east-1.amazonaws.com controlplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS
USA Ouest (Oregon)	us-west-2	controlplane.payment-cryptography.us-west-2.amazonaws.com controlplane.payment-cryptography.us-west-2.api.aws	HTTPS HTTPS
Afrique (Le Cap)	af-south-1	controlplane.payment-cryptography.af-south-1.amazonaws.com controlplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS
Asie-Pacifique (Hyderabad)	ap-south-2	controlplane.payment-cryptography.ap-south-2.amazonaws.com controlplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS

Nom de la région	Région	Point de terminaison	Protocole
Asia Pacific (Mumbai)	ap-south-1	controlplane.payment-cryptography.ap-south-1.amazonaws.com controlplane.payment-cryptography.ap-south-1.api.aws	HTTPS HTTPS
Asie-Pacifique (Osaka)	ap-northeast-3	controlplane.payment-cryptography.ap-northeast-3.amazonaws.com controlplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS
Asie-Pacifique (Singapour)	ap-southeast-1	controlplane.payment-cryptography.ap-southeast-1.amazonaws.com controlplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS
Asie-Pacifique (Sydney)	ap-southeast-2	controlplane.payment-cryptography.ap-southeast-2.amazonaws.com controlplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS
Asie-Pacifique (Tokyo)	ap-northeast-1	controlplane.payment-cryptography.ap-northeast-1.amazonaws.com controlplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS
Canada (Centre)	ca-central-1	controlplane.payment-cryptography.ca-central-1.amazonaws.com controlplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS

Nom de la région	Région	Point de terminaison	Protocole
Europe (Francfort)	eu-central-1	controlplane.payment-cryptography.eu-central-1.amazonaws.com controlplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS
Europe (Irlande)	eu-west-1	controlplane.payment-cryptography.eu-west-1.amazonaws.com controlplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS
Europe (Londres)	eu-west-2	controlplane.payment-cryptography.eu-west-2.amazonaws.com controlplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS
Europe (Paris)	eu-west-3	controlplane.payment-cryptography.eu-west-3.amazonaws.com controlplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS
Amérique du Sud (São Paulo)	sa-east-1	controlplane.payment-cryptography.sa-east-1.amazonaws.com controlplane.payment-cryptography.sa-east-1.api.aws	HTTPS HTTPS

Points de terminaison du plan de données

Nom de la région	Région	Point de terminaison	Protocole	
US East (Ohio)	us-east-2	dataplane.payment-cryptography.us-east-2.amazonaws.com dataplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS	
USA Est (Virginie du Nord)	us-east-1	dataplane.payment-cryptography.us-east-1.amazonaws.com dataplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS	
USA Ouest (Oregon)	us-west-2	dataplane.payment-cryptography.us-west-2.amazonaws.com dataplane.payment-cryptography.us-west-2.api.aws	HTTPS HTTPS	
Afrique (Le Cap)	af-south-1	dataplane.payment-cryptography.af-south-1.amazonaws.com dataplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS	
Asie-Pacifique (Hyderabad)	ap-south-2	dataplane.payment-cryptography.ap-south-2.amazonaws.com dataplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS	
Asia Pacific (Mumbai)	ap-south-1	dataplane.payment-cryptography.ap-south-1.amazonaws.com	HTTPS HTTPS	

Nom de la région	Région	Point de terminaison	Protocole
		dataplane.payment-cryptography.ap-south-1.api.aws	
Asie-Pacifique (Osaka)	ap-northeast-3	dataplane.payment-cryptography.ap-northeast-3.amazonaws.com dataplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS
Asie-Pacifique (Singapour)	ap-southeast-1	dataplane.payment-cryptography.ap-southeast-1.amazonaws.com dataplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS
Asie-Pacifique (Sydney)	ap-southeast-2	dataplane.payment-cryptography.ap-southeast-2.amazonaws.com dataplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS
Asie-Pacifique (Tokyo)	ap-northeast-1	dataplane.payment-cryptography.ap-northeast-1.amazonaws.com dataplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS
Canada (Centre)	ca-central-1	dataplane.payment-cryptography.ca-central-1.amazonaws.com dataplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS

Nom de la région	Région	Point de terminaison	Protocole
Europe (Francfort)	eu-central-1	dataplane.payment-cryptography.eu-central-1.amazonaws.com dataplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS
Europe (Irlande)	eu-west-1	dataplane.payment-cryptography.eu-west-1.amazonaws.com dataplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS
Europe (Londres)	eu-west-2	dataplane.payment-cryptography.eu-west-2.amazonaws.com dataplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS
Europe (Paris)	eu-west-3	dataplane.payment-cryptography.eu-west-3.amazonaws.com dataplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS
Amérique du Sud (São Paulo)	sa-east-1	dataplane.payment-cryptography.sa-east-1.amazonaws.com dataplane.payment-cryptography.sa-east-1.api.aws	HTTPS HTTPS

Débuter avec la cryptographie des AWS paiements

Pour commencer à utiliser le chiffrement des AWS paiements, vous devez d'abord créer des clés, puis les utiliser dans diverses opérations cryptographiques. Le didacticiel ci-dessous fournit un cas d'utilisation simple de génération d'une clé à utiliser pour generating/verifying CVV2 les valeurs. Pour essayer d'autres exemples et explorer les modèles de déploiement au sein d'AWS, essayez l'atelier de cryptographie des AWS paiements suivant ou explorez notre exemple de projet disponible sur GitHub

Ce didacticiel explique comment créer une clé unique et effectuer des opérations cryptographiques à l'aide de cette clé. Ensuite, vous supprimez la clé si vous ne la souhaitez plus, ce qui complète le cycle de vie de la clé.

> **Warning**
>
> Les exemples présentés dans ce guide de l'utilisateur peuvent utiliser des valeurs d'échantillon. Nous vous recommandons vivement de ne pas utiliser de valeurs d'échantillon dans un environnement de production, telles que les numéros de série clés.

Rubriques

Conditions préalables

Avant de commencer, assurez-vous que :

- Vous êtes autorisé à accéder au service. Pour plus d'informations, consultez la section Politiques IAM.

- Vous l'avez AWS CLIinstallé. Vous pouvez également utiliser AWS SDKsou accéder AWS APIsà la cryptographie des AWS paiements, mais les instructions de ce didacticiel utilisent le AWS CLI.

Étape 1 : Création d'une clé

La première étape consiste à créer une clé. Dans ce didacticiel, vous allez créer une clé CVK 3DES double longueur (2KEY TDES) pour générer et vérifier les valeurs CVV/. CVV2

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
        "KeyAttributes": {
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_2KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "CADDA1",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
```

```
        "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
    }
}
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east- 2:111122223333:key/tqv5yij6wtxx64pi. Vous en aurez besoin à l'étape suivante.

Étape 2 : générer une CVV2 valeur à l'aide de la clé

Dans cette étape, vous générez un CVV2 pour une date donnée PAN et une date d'expiration à l'aide de la clé de l'étape 1.

```
$  aws payment-cryptography-data generate-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --primary-account-number=171234567890123 \
    --generation-attributes CardVerificationValue2={CardExpiryDate=0123}
```

```
{
    "CardDataGenerationKeyCheckValue": "CADDA1",
    "CardDataGenerationKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi",
    "CardDataType": "CARD_VERIFICATION_VALUE_2",
    "CardDataValue": "144"
}
```

Prenez note du `cardDataValue` numéro 144 à 3 chiffres dans ce cas. Vous en aurez besoin à l'étape suivante.

Étape 3 : Vérifiez la valeur générée à l'étape 2

Dans cet exemple, vous validez le formulaire CVV2 de l'étape 2 à l'aide de la clé que vous avez créée à l'étape 1.

Exécutez la commande suivante pour valider le CVV2.

```
$  aws payment-cryptography-data  verify-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --primary-account-number=171234567890123 \
```

```
    --verification-attributes CardVerificationValue2={CardExpiryDate=0123} \
    --validation-data 144
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1"
}
```

Le service renvoie une réponse HTTP de 200 pour indiquer qu'il a validé le CVV2.

Étape 4 : Réaliser un test négatif

Au cours de cette étape, vous créez un test négatif dans lequel le résultat n' CVV2 est pas correct et n'est pas validé. Vous essayez de valider une erreur à l' CVV2 aide de la clé que vous avez créée à l'étape 1. Il s'agit d'une opération attendue, par exemple si le titulaire de la carte a mal saisi CVV2 la commande.

```
$  aws payment-cryptography-data  verify-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --primary-account-number=171234567890123 \
    --verification-attributes CardVerificationValue2={CardExpiryDate=0123} \
    --validation-data 999
```

```
Card validation data verification failed.
```

Le service renvoie une réponse HTTP de 400 avec le message « Échec de la vérification des données de validation de la carte » et le motif INVALID_VALIDATION_DATA.

Étape 5 : (Facultatif) Nettoyer

Vous pouvez maintenant supprimer la clé que vous avez créée à l'étape 1. Pour minimiser les modifications irrécupérables, la période de suppression des clés par défaut est de sept jours.

```
$ aws payment-cryptography delete-key \
    --key-identifier=arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi
```

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T08:27:51.795000-07:00",
        "DeletePendingTimestamp": "2022-11-03T13:37:12.114000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "CADDA1",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "DELETE_PENDING",
        "UsageStartTimestamp": "2022-10-27T08:27:51.753000-07:00"
    }
}
```


Prenez note de deux champs dans le résultat. La valeur `deletePendingTimestamp` est fixée par défaut à sept jours dans le futur. Le KeyState est défini sur. `DELETE_PENDING` Vous pouvez annuler cette suppression à tout moment avant l'heure de suppression prévue en appelant `restore-key`.

Gestion des clés

Pour commencer à utiliser la cryptographie des AWS paiements, créez une clé de cryptographie des AWS paiements.

Cette section explique comment créer et gérer différents types de clés de chiffrement des AWS paiements tout au long de leur cycle de vie. Vous apprendrez à créer, afficher et modifier des clés, ainsi qu'à étiqueter des clés, à créer des alias de clé et à activer ou désactiver des clés.

Une clé AWS de cryptographie de paiement est une ressource régionale. Si vous avez l'intention d'utiliser une clé donnée à plusieurs reprises Régions AWS, vous pouvez activer la réplication des Multi-Region clés qui copie en toute sécurité le matériel clé et les métadonnées Régions AWS que vous spécifiez au sein de la même AWS partition et du même compte. La clé source utilisée pour la réplication des Multi-Region clés est connue sous le nom de clé de région primaire (PRK) et elle reste la source faisant autorité pour toutes les activités de gestion clés. La clé répliquée est connue sous le nom de clé Replica Region (RRK) et il s'agit d'une réplique en lecture seule du PRK. Vous devriez envisager d'utiliser Multi-Region des clés avec vos clés pour atteindre les objectifs de conception en matière de disponibilité, de reprise après sinistre et de faible latence.

Rubriques

Création de clés

Vous pouvez créer des clés AWS de cryptographie de paiement à l'aide de l'opération CreateKey API. Lorsque vous créez une clé, vous spécifiez des attributs tels que l'algorithme de clé, l'utilisation

de la clé, les opérations autorisées et si elle est exportable. Vous ne pouvez pas modifier ces propriétés après avoir créé la clé AWS de chiffrement des paiements.

> **Note**
>
> Si la réplication de Multi-Region clé est activée pour vous Compte AWS et que vous créez une clé de cryptographie de paiement, cette clé deviendra automatiquement une clé de région primaire (PRK). Le PRK est répliqué même si vous ne spécifiez pas le `--replication-regions` paramètre dans la CreateKey commande. Pour de plus amples informations, veuillez consulter Comment fonctionne Multi-Region la réplication des clés.

Exemples

- Création d'une clé de dérivation de base 3KEY TDES
- Création d'une clé TDES 2KEY pour CVV/CVV2
- Création d'une clé HMAC
- Création d'une AES-256 clé
- Création d'une clé de chiffrement par code PIN (PEK)
- Création d'une clé asymétrique (RSA)
- Création d'une clé de vérification par code PIN (PVV)
- Création d'une clé ECC asymétrique

Création d'une clé de dérivation de base 3KEY TDES

Example

Cette commande crée une clé de dérivation TDES à 3 touches qui sera répliquée dans les régions USA Est (Ohio) et USA Ouest (Oregon). La réponse inclut les paramètres de demande, un Amazon Resource Name (ARN) pour les appels suivants et une valeur de vérification clé (KCV).

```
$ aws payment-cryptography create-key --exportable --key-attributes \
     "KeyUsage=TR31_B0_BASE_DERIVATION_KEY, \
     KeyClass=SYMMETRIC_KEY,KeyAlgorithm=TDES_3KEY, \
     KeyModesOfUse={NoRestrictions=true}" \
     --replication-regions us-east-2 --region us-west-2
```

Exemple de sortie :


```
{
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "FE23D3",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": true,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_B0_BASE_DERIVATION_KEY"
        },
        "KeyCheckValue": "FE23D3",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-26T16:04:11.559000-07:00"
}
```

Création d'une clé TDES 2KEY pour CVV/CVV2

Example

Cette commande crée une clé TDES 2KEY pour générer et vérifier CVV/CVV2 des valeurs. La réponse inclut les paramètres de la demande, un Amazon Resource Name (ARN) pour les appels suivants et une Key Check Value (KCV).

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY, \
    KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY, \
    KeyModesOfUse='{Generate=true,Verify=true}'
```

Exemple de sortie :

```
{
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "AEA5CD",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-26T16:04:11.559000-07:00"
    }
}
```

Création d'une clé HMAC

Example

Les clés HMAC sont utilisées pour générer ou vérifier les codes d'authentification par message de hachage (HMAC). Avec les clés HMAC, le type de hachage est attribué au moment de la création de la clé (comme HMAC_SHA224 et HMAC_SHA512) et ne peut pas être modifié.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=HMAC_SHA512,KeyUsage=TR31_M7_HMAC_KEY,KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Gene
 = true,Verify = true}'
```

Exemple de sortie :

```
{
 "Key": {
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/qnobl5lghrzunce6",
 "KeyAttributes": {
 "KeyUsage": "TR31_M7_HMAC_KEY",
 "KeyClass": "SYMMETRIC_KEY",
 "KeyAlgorithm": "HMAC_SHA512",
 "KeyModesOfUse": {
 "Encrypt": false,
 "Decrypt": false,
 "Wrap": false,
 "Unwrap": false,
 "Generate": true,
 "Sign": false,
 "Verify": true,
 "DeriveKey": false,
 "NoRestrictions": false
 }
 },
 "KeyCheckValue": "2976E7",
 "KeyCheckValueAlgorithm": "HMAC",
 "Enabled": true,
 "Exportable": true,
 "KeyState": "CREATE_COMPLETE",
 "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
 "CreateTimestamp": "2025-07-30T10:06:12.142000-07:00",
 "UsageStartTimestamp": "2025-07-30T10:06:12.128000-07:00"
 }
}
```

Création d'une AES-256 clé

Example

Cette commande crée une clé AES-256 symétrique pour le chiffrement et le déchiffrement des données. Les clés AES fournissent un cryptage puissant pour les données sensibles et sont couramment utilisées dans le traitement des paiements pour crypter les données des titulaires de cartes et d'autres informations sensibles, mais le TDES est plus couramment utilisé pour les cas d'utilisation par les émetteurs tels que l'EMV.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=AES_256,KeyUsage=TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY,KeyClass=SYMMETRIC_KEY,Key
```

Exemple de sortie :

```
{
    "Key": {
        "CreateTimestamp": "2025-02-02T10:15:30.142000-08:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "AES_256",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY"
        },
        "KeyCheckValue": "2976F5",
        "KeyCheckValueAlgorithm": "CMAC",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2025-02-02T10:15:30.128000-08:00"
    }
}
```

Création d'une clé de chiffrement par code PIN (PEK)

Example

Cette commande crée une clé TDES à 3 touches pour chiffrer les valeurs PIN, bien que les clés PIN puissent également être AES en fonction de vos besoins en matière d'interopérabilité. Vous pouvez utiliser cette clé pour stocker des codes PIN en toute sécurité ou les déchiffrer lors de la vérification, par exemple lors d'une transaction. La réponse inclut les paramètres de demande, un ARN pour les appels suivants et un KCV.

```
$ aws payment-cryptography create-key --exportable --key-attributes \
    KeyAlgorithm=TDES_3KEY,KeyUsage=TR31_P0_PIN_ENCRYPTION_KEY, \

 KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Encrypt=true,Decrypt=true,Wrap=true,Unwrap=true}'
```

Exemple de sortie :

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T08:27:51.795000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY"
        },
        "KeyCheckValue": "7CC9E2",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T08:27:51.753000-07:00"
    }
}
```

Création d'une clé asymétrique (RSA)

Example

Cette commande génère une nouvelle paire de clés asymétriques RSA 2048 bits. Il crée une nouvelle clé privée et la clé publique correspondante. Vous pouvez récupérer la clé publique à l'aide de l'PublicCertificateAPI get.

```
$ aws payment-cryptography create-key --exportable \
    --key-attributes
 KeyAlgorithm=RSA_2048,KeyUsage=TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION, \
    KeyClass=ASYMMETRIC_KEY_PAIR,KeyModesOfUse='{Encrypt=true,
 Decrypt=True,Wrap=True,Unwrap=True}'
```

Exemple de sortie :

```
{
    "Key": {
        "CreateTimestamp": "2022-11-15T11:15:42.358000-08:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
nsq2i3mbg6sn775f",
        "KeyAttributes": {
            "KeyAlgorithm": "RSA_2048",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION"
        },
        "KeyCheckValue": "40AD487F",
        "KeyCheckValueAlgorithm": "SHA-1",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-11-15T11:15:42.182000-08:00"
    }
}
```

Création d'une clé de vérification par code PIN (PVV)

Example

Cette commande crée une clé TDES 3KEY pour générer des valeurs PVV. Vous pouvez utiliser cette clé pour générer un PVV qui peut être comparé à un PVV calculé ultérieurement. La réponse inclut les paramètres de demande, un ARN pour les appels suivants et un KCV.

```
$ aws payment-cryptography create-key --exportable \
    --key-attributes KeyAlgorithm=TDES_3KEY,KeyUsage=TR31_V2_VISA_PIN_VERIFICATION_KEY,
 \
    KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Generate=true,Verify=true}'
```

Exemple de sortie :

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T10:22:59.668000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY"
        },
        "KeyCheckValue": "7F2363",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T10:22:59.614000-07:00"
    }
}
```

Création d'une clé ECC asymétrique

Example

Cette commande génère une paire de clés ECC pour établir un accord de clé ECDH (courbe elliptique Diffie-Hellman) entre deux parties. Avec l'ECDH, chaque partie génère sa propre paire de clés ECC avec l'objectif K3 et le mode d'utilisation X, et ils échangent des clés publiques. Les deux parties utilisent ensuite leur clé privée et la clé publique reçue pour établir une clé dérivée partagée. Pour maintenir le principe d'usage unique des clés cryptographiques dans les paiements, nous recommandons de ne pas réutiliser les paires de clés ECC à des fins multiples, telles que la dérivation et la signature de clés ECDH.

```
$ aws payment-cryptography create-key --exportable \
    --key-attributes
 KeyAlgorithm=ECC_NIST_P256,KeyUsage=TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT, \
    KeyClass=ASYMMETRIC_KEY_PAIR,KeyModesOfUse='{DeriveKey=true}'
```

Exemple de sortie :

```
{
    "Key": {
        "CreateTimestamp": "2024-10-17T01:31:55.908000+00:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
wc3rjsssguhxtilv",
        "KeyAttributes": {
            "KeyAlgorithm": "ECC_NIST_P256",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": true,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": false,
                "Wrap": false
            },
            "KeyUsage": "TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT"
        },
        "KeyCheckValue": "7E34F19F",
        "KeyCheckValueAlgorithm": "SHA-1",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2024-10-17T01:31:55.866000+00:00"
    }
}
```

Clés de listage

Utilisez cette ListKeys opération pour obtenir une liste des clés auxquelles vous pouvez accéder dans votre compte et dans votre région.

Example

```
$ aws payment-cryptography list-keys
```

Exemple de sortie :

```
{
    "Keys": [
        {
            "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
            "Enabled": false,
            "Exportable": true,
            "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "KeyAttributes": {
                "KeyAlgorithm": "TDES_3KEY",
                "KeyClass": "SYMMETRIC_KEY",
                "KeyModesOfUse": {
                    "Decrypt": true,
                    "DeriveKey": false,
                    "Encrypt": true,
                    "Generate": false,
                    "NoRestrictions": false,
                    "Sign": false,
                    "Unwrap": true,
                    "Verify": false,
                    "Wrap": true
                },
                "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
            },
            "KeyCheckValue": "7F2363",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "KeyState": "CREATE_COMPLETE",
            "UsageStopTimestamp": "2022-10-27T14:19:42.488000-07:00"
        }
    ]
}
```

Activation et désactivation de clés

Vous pouvez désactiver et réactiver les clés AWS de chiffrement des paiements. Lorsque vous créez une clé, elle est activée par défaut. Si vous désactivez une clé, elle ne peut être utilisée dans aucune opération cryptographique tant que vous ne l'avez pas réactivée. Start/stop les commandes d'utilisation prennent effet immédiatement, il est donc recommandé de vérifier l'utilisation avant de procéder à une telle modification. Vous pouvez également définir une modification (démarrer ou arrêter l'utilisation) pour qu'elle prenne effet dans le futur à l'aide du `timestamp` paramètre optionnel.

Comme elle est temporaire et facile à annuler, la désactivation d'une clé de chiffrement de AWS paiement constitue une alternative plus sûre à la suppression d'une clé de cryptographie de AWS paiement, une action destructrice et irréversible. Si vous envisagez de supprimer une clé de chiffrement des AWS paiements, désactivez-la d'abord et assurez-vous que vous n'aurez pas besoin de l'utiliser pour chiffrer ou déchiffrer des données à l'avenir.

Rubriques

Commencer à utiliser les clés

L'utilisation des clés doit être activée afin d'utiliser une clé pour les opérations cryptographiques. Si une clé n'est pas activée, vous pouvez utiliser cette opération pour la rendre utilisable. Le champ `UsageStartTimestamp` indiquera le moment où la clé became/will sera active. Ce sera le cas dans le passé pour un jeton activé, et dans le futur s'il est en attente d'activation.

Example

Dans cet exemple, il est demandé qu'une clé soit activée pour son utilisation. La réponse inclut les informations clés et l'indicateur d'activation est passé à vrai. Cela se reflétera également dans l'objet de réponse list-keys.

```
$ aws payment-cryptography start-key-usage --key-identifier "arn:aws:payment-
cryptography:us-east-2:111122223333:key/alsuwfxug3pgy6xh"
```

```
{
      "Key": {
      "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
      "Enabled": true,
      "Exportable": true,
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
      "KeyAttributes": {
          "KeyAlgorithm": "TDES_3KEY",
          "KeyClass": "SYMMETRIC_KEY",
          "KeyModesOfUse": {
              "Decrypt": true,
              "DeriveKey": false,
              "Encrypt": true,
              "Generate": false,
              "NoRestrictions": false,
              "Sign": false,
              "Unwrap": true,
              "Verify": false,
              "Wrap": true
          },
          "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
      },
      "KeyCheckValue": "369D",
      "KeyCheckValueAlgorithm": "ANSI_X9_24",
      "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
      "KeyState": "CREATE_COMPLETE",
      "UsageStartTimestamp": "2022-10-27T14:09:59.468000-07:00"
  }
}
```

Arrêter l'utilisation des clés

Si vous n'avez plus l'intention d'utiliser une clé, vous pouvez arrêter de l'utiliser pour empêcher de nouvelles opérations cryptographiques. Cette opération n'est pas permanente, vous pouvez donc l'inverser en utilisant la clé de départ. Vous pouvez également définir une clé pour qu'elle soit désactivée à l'avenir. Le champ `UsageStopTimestamp` indiquera le moment où la clé became/will sera désactivée.

Example

Dans cet exemple, il est demandé d'arrêter l'utilisation des clés à l'avenir. Après exécution, cette clé ne peut pas être utilisée pour des opérations cryptographiques à moins d'être réactivée via l'utilisation de la clé de démarrage. La réponse inclut les informations relatives à la clé et l'indicateur d'activation est passé à faux. Cela se reflétera également dans l'objet de réponse list-keys.

```
$ aws payment-cryptography stop-key-usage --key-identifier "arn:aws:payment-
cryptography:us-east-2:111122223333:key/alsuwfxug3pgy6xh"
```

```
{
  "Key": {
      "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
      "Enabled": false,
      "Exportable": true,
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
      "KeyAttributes": {
          "KeyAlgorithm": "TDES_3KEY",
          "KeyClass": "SYMMETRIC_KEY",
          "KeyModesOfUse": {
              "Decrypt": true,
              "DeriveKey": false,
              "Encrypt": true,
              "Generate": false,
              "NoRestrictions": false,
              "Sign": false,
              "Unwrap": true,
              "Verify": false,
              "Wrap": true
          },
          "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
      },
      "KeyCheckValue": "369D",
      "KeyCheckValueAlgorithm": "ANSI_X9_24",
      "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
      "KeyState": "CREATE_COMPLETE",
      "UsageStopTimestamp": "2022-10-27T14:09:59.468000-07:00"
  }
}
```

Réplication des clés AWS de chiffrement des paiements

AWS La cryptographie des paiements prend en charge la réplication des Multi-Region clés, ce qui vous permet de distribuer en toute sécurité le matériel clé et les métadonnées d'une clé de chiffrement de AWS paiement donnée vers une ou plusieurs personnes Régions AWS au sein de la même AWS partition et du même compte.

La clé source est connue sous le nom de clé de région primaire (PRK) et reste la source officielle pour toutes les activités de gestion des clés, tandis que les clés PRK et les clés de région de réplication (RRK) peuvent être utilisées pour les opérations cryptographiques dans leurs domaines respectifs. Régions AWS

Avantages de la réplication Multi-Region des clés

Voici quelques avantages de la réplication des Multi-Region clés.

- Configuration simplifiée pour les applications à haute disponibilité - AWS Payment Cryptography gère la distribution des clés pour vous afin que vous puissiez utiliser une clé en plusieurs fois Régions AWS sans avoir à créer de copies découplées d'une clé donnée.
- Clés à haute disponibilité et à faible latence - Grâce à la réplication des Multi-Region clés, vous pouvez accéder à vos clés de manière multiple, Régions AWS ce qui les rend hautement disponibles, ce qui réduit le temps de latence.
- Durabilité du matériau clé - Les clés de région de réplique sont des répliques complètes de clés et peuvent être utilisées indépendamment de leur clé de région principale dans les opérations cryptographiques. Un RRK fournit une réplique durable en cas de perte de données catastrophique d'un PRK.

Comment fonctionne Multi-Region la réplication des clés

Lorsque la réplication des Multi-Region clés est activée, le service de cryptographie des AWS paiements utilise des mécanismes de distribution de clés sécurisés pour copier les informations clés et les métadonnées vers la réplique Régions AWS que vous spécifiez. Les modifications apportées aux métadonnées clés d'une région principale, telles que les attributs clés, l'état et l'activation, sont automatiquement répliquées sur les clés de la région de réplication.

Limites et considérations

Voici certaines des Multi-Region principales limites et considérations relatives à la réplication.

- Vous devez activer cette fonctionnalité pour une clé de cryptographie de paiement Région AWS ou pour des clés spécifiques.
  - Si cette fonctionnalité est activée pour un Région AWS, toutes les clés de chiffrement des AWS paiements créées après l'activation seront répliquées à la valeur spécifiée. Région AWS Les clés créées dans cette région deviendront des clés de région principale. Les clés existantes dans cette région ne seront pas automatiquement répliquées. Vous pouvez activer la réplication Multi-Region des clés pour les clés existantes au niveau de la clé ou Région AWS au niveau de celle-ci.
  - Chacun Région AWS peut avoir des paramètres de réplication Multi-Region clés uniques.
  - Les paramètres de Multi-Region réplication d'une clé ont priorité sur les paramètres de réplication de la Région AWS Multi-Region clé.
- Une clé de région de réplication ne peut pas être configurée pour être répliquée vers une autre Régions AWS.
- Multi-Region la réplication de clés est disponible pour les clés de cryptographie de paiement symétriques telles que Triple Data Encryption Standard (3DES), Advanced Encryption Standard (AES) et Hash-based Message Authentication Code (HMAC).
- Les clés de cryptographie de paiement asymétriques ne prennent pas en charge la réplication des Multi-Region clés.
- Les clés Replica Region sont des clés en lecture seule. Toutes les modifications apportées à la clé de région principale seront appliquées aux clés de région de réplication.
- Les modifications des clés de région principale sont finalement cohérentes avec les clés de région de réplication.
- Les clés de chiffrement des paiements ne peuvent être répliquées qu'avec la même AWS partition et le même compte.
- Le nombre de clés de la région de réplication est pris en compte dans la limite AWS de cryptographie des paiements de votre Compte AWS niveau.
- La clé de région principale et la clé de région de réplication utilisent le même identifiant de clé, ce qui vous permet de référencer les deux clés par le même ARN dans les politiques IAM.
- Pour que la réplication réussisse, vous devez disposer Région AWS d'`CreateKey`autorisations sur le réplica.

Activation de Multi-Region la réplication des clés

Vous pouvez activer la réplication des clés pour les Multi-Region clés de cryptographie des AWS paiements de deux manières.

1. Région AWS: Multi-Region la réplication des clés est appliquée à toutes les nouvelles clés créées Région AWS lorsqu'elle est activée. Cette méthode assure une réplication cohérente pour toutes les clés.
2. Clés AWS de cryptographie de paiement spécifiques : vous pouvez gérer la réplication des Multi-Region clés pour des clés individuelles, ce qui permet un niveau de contrôle plus précis.

Une fois la réplication des Multi-Region clés activée, vos clés de chiffrement des paiements seront répliquées selon Régions AWS vos spécifications.

> **Important**
>
> Multi-Region la réplication des clés ne peut pas être interrompue. Vos clés sont automatiquement répliquées vers celles Régions AWS que vous spécifiez une fois la réplication activée. Multi-Region la réplication des clés peut être désactivée pour une clé spécifique Région AWS ou pour les clés de cryptographie des paiements. Vous devez supprimer la clé de région principale en Région AWS tant que région de réplication pour supprimer la clé de région de réplication.
> Vous pouvez également appeler la commande `StopKeyUsage`API ou stop-key-usageCLI sur votre PRK pour arrêter l'utilisation du PRK et de tous les RRK associés. Vous ne pourrez pas utiliser ces clés dans des opérations cryptographiques. L'utilisation de l'`StopKeyUsage`API ou de la commande stop-key-usage CLI n'arrêtera pas la réplication de Multi-Region clé en cours activée pour votre PRK.

Vous pouvez vérifier les paramètres de réplication des Multi-Region clés pour les clés de chiffrement des AWS paiements dans un domaine spécifique en Région AWS appelant l'`GetDefaultKeyReplicationRegions`API ou la commande get-default-key-replication-regions CLI. Les touches à l' Région AWS endroit où vous appelez cette action ou commande d'API deviendront votre PRK.

Utilisez les procédures suivantes pour activer la réplication Multi-Region des clés.

For Région AWS

- Utilisez la commande suivante pour activer la réplication Multi-Region des clés pour un Région AWS que vous spécifiez. Dans cet exemple, la réplication des Multi-Region clés est activée dans l'est des États-Unis (Ohio) et dans l'ouest des États-Unis (Oregon). Pour utiliser cette commande, remplacez celle *italicized placeholder text* de l'exemple par vos propres informations.

```
aws payment-cryptography enable-default-key-replication-regions \
    --replication-regions us-east-2 us-west-2
```

> **Note**
>
> L'activation de la réplication des Multi-Region clés pour et ne Région AWS modifiera pas la configuration de réplication des clés de cryptographie de AWS paiement existantes. Vous pouvez activer cette fonctionnalité pour les clés existantes au niveau des clés. Seules les clés créées après l'activation de la réplication des Multi-Region clés sont utilisées pour et Région AWS utiliseront les paramètres de réplication des régions.

For specific AWS Payment Cryptography keys

- Utilisez la commande suivante pour activer la réplication de Multi-Region clés pour des clés de chiffrement de paiement spécifiques. Dans cet exemple, la réplication des Multi-Region clés est activée dans l'est des États-Unis (Ohio). Pour utiliser cette commande, remplacez celle *italicized placeholder text* de l'exemple par vos propres informations.

```
aws payment-cryptography add-key-replication-regions \
    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --replication-regions us-east-2
```

Vous pouvez également créer une nouvelle clé de chiffrement des paiements en activant cette fonctionnalité en incluant la réplication Régions AWS dans votre demande de création de clé.

> **Note**
>
> Les principaux paramètres de réplication ont priorité sur le paramètre de Région AWS réplication.

Désactivation de la réplication Multi-Region des clés

Si vous souhaitez désactiver la réplication des Multi-Region clés, vous pouvez appeler les commandes disable-default-key-replication ou les commandes remove-key-replication-regions CLI, selon le mode d'activation de la réplication des Multi-Region clés. Vous devez spécifier l'ARN de la clé et le Région AWS pour désactiver la réplication de la Multi-Region clé.

Considérations

Les suppressions de clés de région de réplication sont finalement cohérentes.

Vous pouvez vérifier les paramètres de réplication des Multi-Region clés pour les clés de chiffrement des AWS paiements dans un domaine spécifique en Région AWS appelant l'`GetDefaultKeyReplicationRegions`API ou la commande get-default-key-replication-regions CLI.

Utilisez les procédures suivantes pour désactiver la réplication Multi-Region des clés.

For Région AWS

- Utilisez la commande suivante pour désactiver la réplication des Multi-Region clés pour un élément Région AWS que vous spécifiez. Dans cet exemple, la réplication des Multi-Region clés est désactivée dans l'est des États-Unis (Ohio). Pour utiliser cette commande, remplacez celle *`italicized placeholder text`* de l'exemple par vos propres informations.

```
aws payment-cryptography disable-default-key-replication-regions \
    --replication-regions us-east-2
```

For specific AWS Payment Cryptography keys

- Utilisez la commande suivante pour désactiver la réplication de Multi-Region clé pour une clé de cryptographie de paiement spécifique. Dans cet exemple, la réplication des Multi-Region

clés est désactivée dans l'est des États-Unis (Ohio). Pour utiliser cette commande, remplacez celle *italicized placeholder text* de l'exemple par vos propres informations.

```
aws payment-cryptography remove-key-replication-regions \
    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --replication-regions us-east-2
```

Considérations sur la sécurité

Les considérations de sécurité suivantes sont à prendre en compte lors de l'utilisation de la réplication de Multi-Region clés pour vos clés de cryptographie de paiement. Pour de plus amples informations, veuillez consulter Bonnes pratiques de sécurité pour la cryptographie des AWS paiements.

- Limitez le partage de documents clés.
- Respectez le principe du moindre privilège lors de la création de politiques IAM.
- Vous ne pouvez pas modifier la clé Replica Region car il s'agit d'une clé en lecture seule.

Bonnes pratiques

Voici quelques bonnes pratiques relatives à l'utilisation de la réplication de Multi-Region clés avec des clés AWS de cryptographie de paiement.

- Assurez-vous que votre application continue de fonctionner même si la réplication de la Multi-Region clé vers la valeur spécifiée n' Région AWS est pas immédiate. Si vous avez besoin de savoir quand la réplication des Multi-Region clés est terminée, vous pouvez effectuer un suivi à l'aide de l'action GetKeyAPI. Vous pouvez surveiller les principaux événements de réplication avec AWS CloudTrail.
- Testez et mettez en œuvre des processus de déploiement automatisés en cas de basculement d'une région Région AWS à l'autre.

Tarification

Les répliques de clés de région que vous créez à l'aide de AWS Payment Cryptography vous sont facturées. Ces clés sont facturées par Région AWS. Pour obtenir les dernières informations sur les

tarifs de la cryptographie des paiements, consultez la page de tarification AWS de la cryptographie des paiements.

Suppression de clés

La suppression d'une clé de chiffrement de AWS paiement supprime le contenu de la clé et toutes les métadonnées associées à la clé et est irréversible, sauf si une copie de la clé est disponible en dehors de la cryptographie des AWS paiements. Une fois qu'une clé est supprimée, vous ne pouvez plus déchiffrer les données chiffrées sous cette clé, ce qui signifie que les données peuvent devenir irrécupérables. Vous ne devez supprimer une clé que lorsque vous êtes certain de ne plus avoir besoin de l'utiliser et qu'aucune autre personne n'utilise cette clé. En cas de doute, pensez à arrêter l'utilisation des clés au lieu de les supprimer. Vous pouvez réactiver une clé désactivée si vous devez la réutiliser ultérieurement, mais vous ne pouvez pas récupérer une clé de chiffrement des AWS paiements supprimée à moins de pouvoir la réimporter depuis une autre source.

Avant de supprimer une clé, assurez-vous que vous n'en avez plus besoin. AWS La cryptographie des paiements ne stocke pas les résultats d'opérations cryptographiques telles que le CVV2 et n'est pas en mesure de déterminer si une clé est nécessaire pour un matériel cryptographique persistant.

AWS La cryptographie des paiements ne supprime jamais les clés appartenant à AWS des comptes actifs, sauf si vous planifiez explicitement leur suppression et que le délai d'attente obligatoire expire.

Toutefois, vous pouvez choisir de supprimer une clé AWS de chiffrement des paiements pour une ou plusieurs des raisons suivantes :

- Pour terminer le cycle de vie d'une clé dont vous n'avez plus besoin
- Pour éviter les frais de gestion associés à la gestion des clés de chiffrement des AWS paiements non utilisées

> **Note**
>
> Si vous fermez ou supprimez votre Compte AWS, votre clé AWS de cryptographie de paiement devient inaccessible. Il n'est pas nécessaire de planifier la suppression de votre clé de chiffrement des AWS paiements indépendamment de la fermeture du compte.

AWS Payment Cryptography enregistre une entrée dans votre AWS CloudTrailjournal lorsque vous planifiez la suppression de la clé cryptographique des AWS paiements et lorsque la clé de cryptographie des AWS paiements est effectivement supprimée.

Lorsque vous utilisez la réplication de Multi-Region clés, lorsque vous supprimez une clé de cryptographie de paiement qui est une clé de région principale (PRK), les clés de région de réplication (RRK) seront également automatiquement supprimées. Un RRK ne peut pas être supprimé comme un PRK. Si vous souhaitez supprimer un RRK, vous devez modifier les régions de réplication de votre PRK.

À propos de la période d'attente

La suppression d'une clé étant irréversible, AWS Payment Cryptography vous oblige à définir un délai d'attente compris entre 3 et 180 jours. Le délai d'attente par défaut est de sept 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 AWS de cryptographie de paiement sera supprimée, utilisez les GetKey opérations. Assurez-vous de noter le fuseau horaire.

Pendant la période d'attente, le statut de la clé AWS de chiffrement des paiements et l'état de la clé sont En attente de suppression.

> **Note**
>
> Une clé AWS de chiffrement de paiement en attente de suppression ne peut être utilisée dans aucune opération cryptographique.

Une fois la période d'attente terminée, AWS Payment Cryptography supprime la clé de cryptographie des AWS paiements, ses alias et toutes les métadonnées associées à AWS la cryptographie des paiements.

Utilisez la période d'attente pour vous assurer que vous n'avez pas besoin de la clé AWS de chiffrement des paiements, ni maintenant ni dans le futur. Si vous constatez que vous avez besoin de la clé pendant la période d'attente, vous pouvez annuler la suppression de la 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 le service supprime la clé.

Example

Dans cet exemple, il est demandé de supprimer une clé. Outre les informations clés de base, deux champs pertinents indiquent que l'état de la clé a été changé en DELETE_PENDING et que delete PendingTimestamp représente le moment où la suppression de la clé est actuellement planifiée.

```
$ aws payment-cryptography delete-key \
                    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

```
  {
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_3KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "0A3674",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": false,
        "Exportable": true,
        "KeyState": "DELETE_PENDING",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-05T12:01:29.969000-07:00",
        "UsageStopTimestamp": "2023-06-05T14:31:13.399000-07:00",
        "DeletePendingTimestamp": "2023-06-12T14:58:32.865000-07:00"
    }
}
```

Example

Dans cet exemple, une suppression en attente est annulée. Une fois l'opération terminée avec succès, aucune clé ne sera supprimée conformément au calendrier précédent. La réponse contient les informations clés de base ; en outre, deux champs pertinents ont été modifiés : `KeyState` et`deletePendingTimestamp`. `KeyState`est renvoyé à la valeur CREATE_COMPLETE, tandis qu'il `DeletePendingTimestamp` est supprimé.

```
$ aws payment-cryptography restore-key --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_3KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "0A3674",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": false,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-08T12:01:29.969000-07:00",
        "UsageStopTimestamp": "2023-06-08T14:31:13.399000-07:00"
    }
}
```

Importation et exportation de clés

Vous pouvez importer des clés AWS de chiffrement de paiement depuis d'autres solutions et les exporter vers d'autres solutions, telles que les HSM. De nombreux clients échangent des clés avec des fournisseurs de services à l'aide des fonctionnalités d'importation et d'exportation. Nous avons conçu la cryptographie des AWS paiements pour utiliser une approche électronique moderne de la gestion des clés qui vous aide à maintenir la conformité et les contrôles. Nous recommandons d'utiliser un échange de clés électronique basé sur des normes plutôt que des composants clés sur papier. Si vous devez continuer à traiter les composants clés en papier jusqu'à ce que tous les partenaires acceptent l'échange électronique de clés, vous pouvez utiliserÉchange de clés physiques.

Points forts minimaux et effet sur les fonctions d'importation et d'exportation

La norme PCI exige des valeurs de clé minimales spécifiques pour les opérations cryptographiques, le stockage des clés et la transmission des clés. Ces exigences peuvent changer lors de la révision des normes PCI. Les règles précisent que les clés d'emballage utilisées pour le stockage ou le transport doivent être au moins aussi solides que la clé à protéger. Nous appliquons cette exigence automatiquement lors de l'exportation et empêchons les clés d'être protégées par des clés plus faibles, comme indiqué dans le tableau suivant.

Le tableau suivant indique les combinaisons de clés d'encapsulage, de clés à protéger et de méthodes de protection prises en charge.

	Clé d'emballage											
La clé de la protection	TDES	TDES	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Remarques
TDES_2KE	TR-3	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	
TDES_3KE	✗ Non pris en charg	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	

	Clé d'emballage											
La clé de la protection	TDE	TDE	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Remarques
AES_128	✗ Non pris en charg	✗ Non pris en charg	TR-3	TR-3	TR-3	✗ Non pris en charg	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	
AES_192	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	TR-3	TR-3	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	ECDI	ECDI	
AES_256	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	TR-3	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	✗ Non pris en charg	ECDI	

Pour plus d'informations, consultez l'annexe D - Tailles et forces de clé minimales et équivalentes pour les algorithmes approuvés dans les normes PCI HSM.

Échange de clé de chiffrement (KEK)

Nous vous recommandons d'utiliser la X9.24 TR-34 norme ANSI. Ce type de clé initial peut être appelé clé de chiffrement (KEK), clé principale de zone (ZMK) ou clé principale de contrôle de zone (ZCMK). Si vos systèmes ou partenaires ne sont pas TR-34 encore pris en charge, vous pouvez utiliser RSA Wrap/Unwrap. Si vos besoins incluent l'échange de AES-256 clés, vous pouvez utiliser l'ECDH.

> **Note**
>
> Pour importer vos propres clés de test ou pour synchroniser les clés avec vos HSM existants, consultez l'exemple de code de cryptographie des AWS paiements sur. GitHub

Échange de clés de travail (WK)

Nous utilisons les normes de l'industrie (ANSI X9.24 TR 31-2018 et X9.143) pour échanger des clés de travail. Cela nécessite que vous ayez déjà échangé un KEK en utilisant TR-34 RSA Wrap, ECDH ou des schémas similaires. Cette approche répond à l'exigence du code PIN PCI qui consiste à lier cryptographiquement le matériel clé à son type et à son utilisation à tout moment. Les clés de travail incluent les clés de travail de l'acquéreur, les clés de travail de l'émetteur, le BDK et l'IPEK.

Rubriques

Clés d'importation

> **Important**
>
> Les exemples nécessitent la dernière version de l'AWS CLI V2. Avant de commencer, assurez-vous d'avoir effectué la mise à niveau vers la dernière version.

Table des matières

Présentation de l'importation de clés

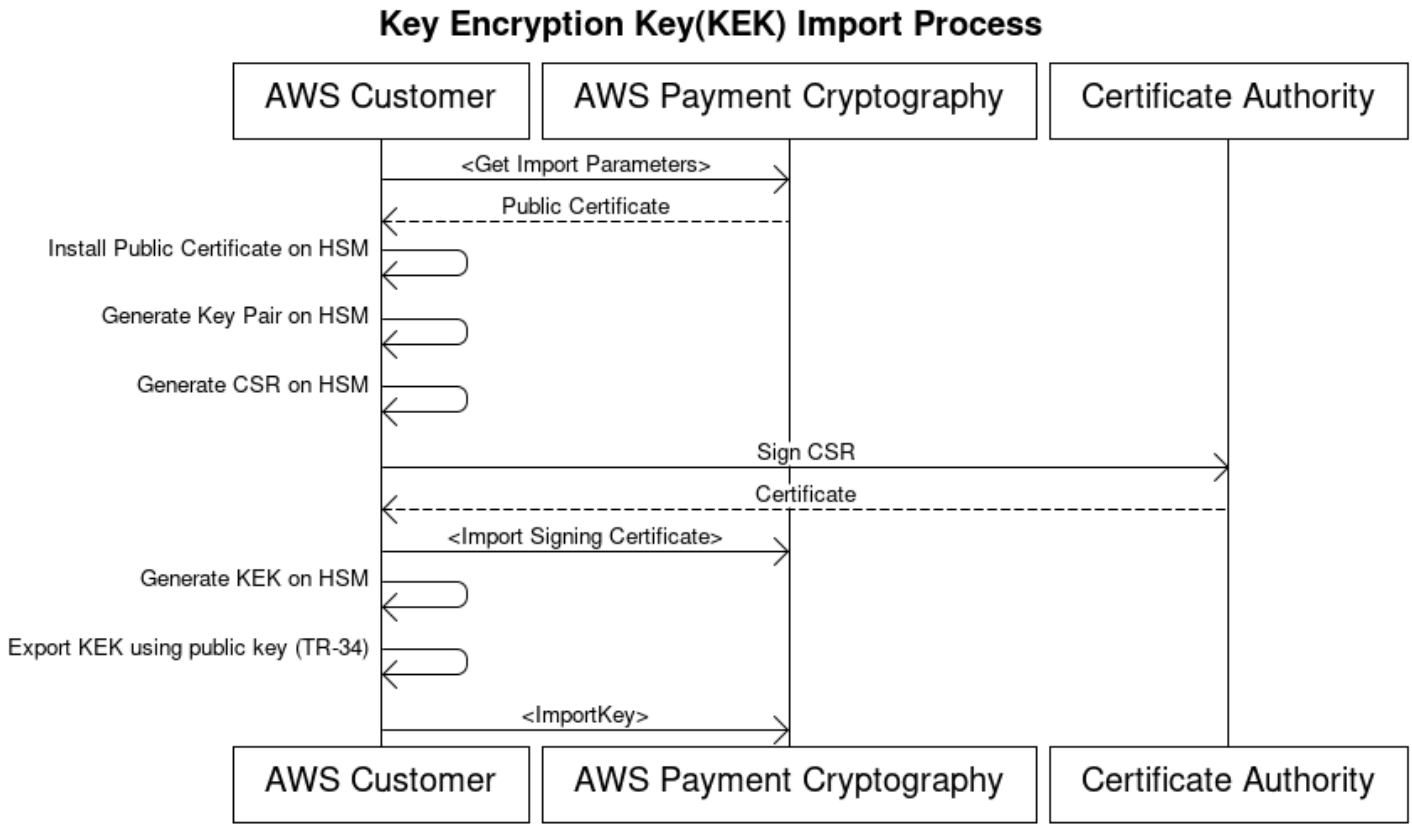
> **Note**
>
> Lorsque vous importez des clés TR-31 ou des blocs de TR-34 clés, AWS Payment Cryptography conserve généralement (mais n'utilise pas) les en-têtes facultatifs. X9.143 L'en-tête HM (type de hachage HMAC) est utilisé lors des opérations cryptographiques. L'en-tête KP (KCV de la clé d'encapsulation) est spécifique au processus d'importation et n'est pas conservé.

Lorsque vous échangez des clés avec une contrepartie, il s'agit généralement d'abord d'échanger une clé d'échange de clés (KEK). Cette clé sera ensuite utilisée pour protéger les clés suivantes. En utilisant des formats électroniques, le KEK peut être échangé à l'aide de techniques asymétriques telles que TR-34 l'ECDH ou le enveloppement RSA. Les clés suivantes seront échangées à l'aide d'un échange de clés symétrique tel que TR-31. Ce KEK aura une longue durée de vie et ne pourra être mis à jour que toutes les quelques années, conformément à la politique et à la période cryptographique définie.

Si seulement une ou deux clés sont échangées, vous pouvez également choisir d'utiliser des techniques asymétriques pour échanger directement cette clé, comme un BDK. AWS La cryptographie des paiements prend en charge les deux méthodes d'échange de clés.

Importation de clés symétriques

Importer des clés à l'aide de techniques asymétriques () TR-34



TR-34 utilise le chiffrement asymétrique RSA pour chiffrer et signer des clés symétriques à des fins d'échange. Cela garantit à la fois la confidentialité (chiffrement) et l'intégrité (signature) de la clé encapsulée.

Pour importer vos propres clés, consultez l'exemple de projet AWS de cryptographie des paiements sur GitHub. Pour obtenir des instructions sur la façon d' import/export accéder aux clés depuis d'autres plateformes, un exemple de code est disponible sur GitHubou consultez le guide de l'utilisateur de ces plateformes.

1. Appelez la commande Initialiser l'importation

   Appelez `get-parameters-for-import` pour initialiser le processus d'importation. Cette API génère une paire de clés pour les importations de clés, signe la clé et renvoie le certificat et la racine du certificat. Chiffrez la clé à exporter à l'aide de cette clé. En TR-34 termes terminologiques, cela s'appelle le certificat KRD. Ces certificats sont codés en base64, ont une durée de vie courte et ne sont destinés qu'à cette fin. Enregistrez la `ImportToken` valeur.

```
$ aws payment-cryptography get-parameters-for-import \
    --key-material-type TR34_KEY_BLOCK \
    --wrapping-key-algorithm RSA_2048
```

```
{
    "ImportToken": "import-token-bwxli6ocftypneu5",
    "ParametersValidUntilTimestamp": 1698245002.065,
    "WrappingKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0....",
    "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0....",
    "WrappingKeyAlgorithm": "RSA_2048"
}
```

2. Installer le certificat public sur le système source clé

   Avec la plupart des HSM, vous devez installer, charger ou approuver le certificat public généré à l'étape 1 pour exporter les clés à l'aide de celui-ci. Cela peut inclure l'ensemble de la chaîne de certificats ou uniquement le certificat racine de l'étape 1, selon le HSM.

3. Générez une paire de clés sur le système source et fournissez une chaîne de certificats à AWS Payment Cryptography

   Pour garantir l'intégrité de la charge utile transmise, l'expéditeur (Key Distribution Host ou KDH) la signe. Générez une clé publique à cette fin et créez un certificat de clé publique (X509) à renvoyer à AWS Payment Cryptography.

   Lorsque vous transférez des clés depuis un HSM, créez une paire de clés sur ce HSM. Le HSM, un tiers ou un service tel que celui-ci AWS CA privée peut générer le certificat.

   Chargez le certificat racine dans AWS Payment Cryptography à l'aide de la `importKey` commande avec KeyMaterialType of `RootCertificatePublicKey` et KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

   Pour les certificats intermédiaires, utilisez la `importKey` commande avec KeyMaterialType of `TrustedCertificatePublicKey` et KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Répétez cette procédure pour plusieurs certificats intermédiaires. Utilisez le dernier certificat importé `KeyArn` de la chaîne comme entrée pour les commandes d'importation suivantes.

> **Note**
>
> N'importez pas le certificat Leaf. Fournissez-le directement lors de la commande d'importation.

4. Exporter la clé depuis le système source

   De nombreux HSM et systèmes associés prennent en charge l'exportation de clés selon la TR-34 norme. Spécifiez la clé publique de l'étape 1 en tant que certificat KRD (chiffrement) et la clé de l'étape 3 en tant que certificat KDH (signature). Pour importer vers AWS Payment Cryptography, spécifiez le format en deux passes TR-34.2012 non CMS, également appelé format TR-34 Diebold.

5. Clé d'importation d'appels

   Appelez l'API ImportKey avec un KeyMaterialType de. `TR34_KEY_BLOCK` Utilisez le KeyARN de la dernière autorité de certification importée à l'étape 3 pour`certificate-authority-public-key-identifier`, le matériel clé encapsulé de l'étape 4 pour `key-material` et le certificat feuille de l'étape 3 pour`signing-key-certificate`. Incluez le jeton d'importation de l'étape 1.

```
$ aws payment-cryptography import-key \
    --key-material='{"Tr34KeyBlock": { \
    "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/zabouwe3574jysdl", \
    "ImportToken": "import-token-bwxli6ocftypneu5", \
    "KeyBlockFormat": "X9_TR34_2012", \
    "SigningKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJ...", \
    "WrappedKeyBlock":
 "308205A106092A864886F70D010702A08205923082058E020101310D300B0609608648016503040201308203.
 \
    }'
```

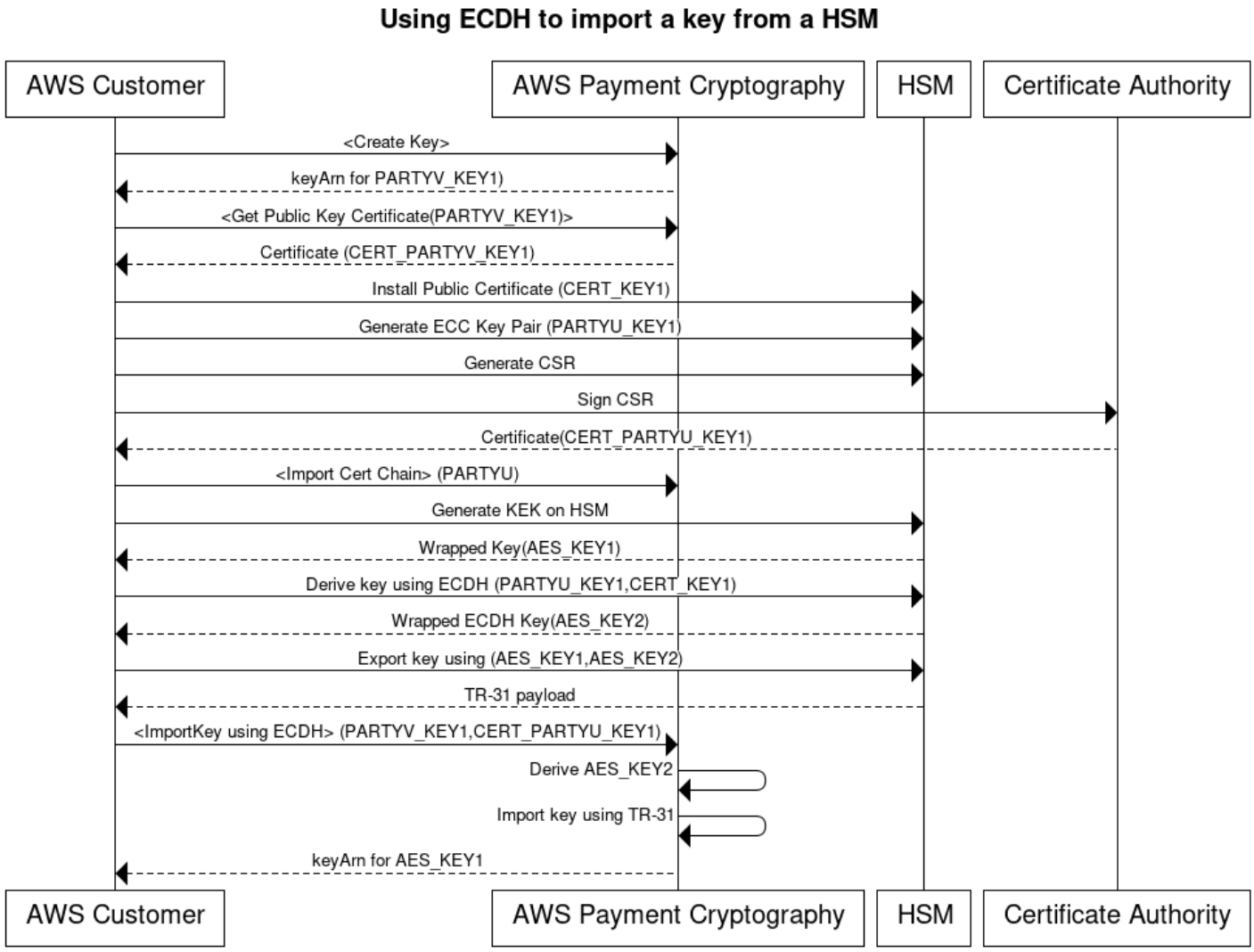
```
{
  "Key": {
    "CreateTimestamp": "2023-06-13T16:52:52.859000-04:00",
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
```

```
    "KeyAttributes": {
      "KeyAlgorithm": "TDES_3KEY",
      "KeyClass": "SYMMETRIC_KEY",
      "KeyModesOfUse": {
        "Decrypt": true,
        "DeriveKey": false,
        "Encrypt": true,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": true,
        "Verify": false,
        "Wrap": true
      },
      "KeyUsage": "TR31_K1_KEY_ENCRYPTION_KEY"
    },
    "KeyCheckValue": "CB94A2",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "KeyOrigin": "EXTERNAL",
    "KeyState": "CREATE_COMPLETE",
    "UsageStartTimestamp": "2023-06-13T16:52:52.859000-04:00"
  }
}
```

6. Utiliser la clé importée pour les opérations cryptographiques ou les importations ultérieures

   Si la clé importée KeyUsage était TR31_K0_KEY_ENCRYPTION_KEY, vous pouvez utiliser cette clé pour les importations de clés suivantes en utilisant. TR-31 Pour les autres types de clés (tels que TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), vous pouvez utiliser la clé directement pour les opérations cryptographiques.

Importation de clés à l'aide de techniques asymétriques (ECDH)



Elliptic Curve Diffie-Hellman (ECDH) utilise la cryptographie asymétrique ECC pour établir une clé partagée entre deux parties sans nécessiter de clés pré-échangées. Les clés ECDH étant éphémères, AWS Payment Cryptography ne les stocke pas. Dans ce processus, une seule fois KBPK/KEKest dérivée à l'aide de l'ECDH. Cette clé dérivée est immédiatement utilisée pour encapsuler la clé que vous souhaitez transférer, qui peut être un autre KBPK, une clé IPEK ou d'autres types de clés.

Lors de l'importation, le système d'envoi est communément appelé Partie U (Initiateur) et la cryptographie des AWS paiements est connue sous le nom de Partie V (Répondeur).

> **Note**
>
> Bien que l'ECDH puisse être utilisé pour échanger n'importe quel type de clé symétrique, il s'agit de la seule approche permettant de transférer AES-256 des clés en toute sécurité.

1. Générer une paire de clés ECC

   Appelez `create-key` pour créer une paire de clés ECC pour ce processus. Cette API génère une paire de clés pour les importations ou les exportations de clés. Lors de la création, spécifiez le type de clés pouvant être dérivées à l'aide de cette clé ECC. Lorsque vous utilisez l'ECDH pour échanger (encapsuler) d'autres clés, utilisez une valeur de`TR31_K1_KEY_BLOCK_PROTECTION_KEY`.

   > **Note**
   >
   > Bien que l'ECDH de bas niveau génère une clé dérivée qui peut être utilisée à n'importe quelle fin, la cryptographie des AWS paiements limite la réutilisation accidentelle d'une clé à des fins multiples en autorisant l'utilisation d'une clé uniquement pour un seul type de clé dérivée.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=ECC_NIST_P256,KeyUsage=TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT,KeyClass=ASYM
 --derive-key-usage "TR31_K1_KEY_BLOCK_PROTECTION_KEY"
```

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv",
                    "KeyAttributes": {
                        "KeyUsage": "TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT",
                        "KeyClass": "ASYMMETRIC_KEY_PAIR",
                        "KeyAlgorithm": "ECC_NIST_P256",
                        "KeyModesOfUse": {
                            "Encrypt": false,
                            "Decrypt": false,
                            "Wrap": false,
                            "Unwrap": false,
```

```
                            "Generate": false,
                            "Sign": false,
                            "Verify": false,
                            "DeriveKey": true,
                            "NoRestrictions": false
                        }
                    },
                    "KeyCheckValue": "2432827F",
                    "KeyCheckValueAlgorithm": "CMAC",
                    "Enabled": true,
                    "Exportable": true,
                    "KeyState": "CREATE_COMPLETE",
                    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                    "CreateTimestamp": "2025-03-28T22:03:41.087000-07:00",
                    "UsageStartTimestamp": "2025-03-28T22:03:41.068000-07:00"
                }
            }
```

2. Obtenir un certificat de clé publique

   Appelez `get-public-key-certificate` pour recevoir la clé publique sous forme de X.509 certificat signé par l'autorité de certification de votre compte, spécifique à la cryptographie des AWS paiements dans une région spécifique.

   Example

```
$ aws payment-cryptography get-public-key-certificate \
                --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv
```

```
{
            "KeyCertificate": "LS0tLS1CRUdJTi...",
            "KeyCertificateChain": "LS0tLS1CRUdJT..."
      }
```

3. Installer un certificat public sur le système de contrepartie (Partie U)

   Avec de nombreux HSM, vous devez installer, charger ou approuver le certificat public généré à l'étape 1 pour exporter les clés à l'aide de celui-ci. Cela peut inclure l'ensemble de la chaîne de certificats ou uniquement le certificat racine de l'étape 1, selon le HSM. Consultez la documentation de votre HSM pour plus d'informations.

4. Générez une paire de clés ECC sur le système source et fournissez une chaîne de certificats à AWS Payment Cryptography

   Dans l'ECDH, chaque partie génère une paire de clés et s'accorde sur une clé commune. Pour que AWS Payment Cryptography puisse dériver la clé, elle a besoin de la clé publique de la contrepartie au format clé X.509 publique.

   Lorsque vous transférez des clés depuis un HSM, créez une paire de clés sur ce HSM. Pour les HSM qui prennent en charge les blocs de touches, l'en-tête de clé ressemblera à`D0144K3EX00E0000`. Lors de la création du certificat, vous générez généralement un CSR sur le HSM, puis le HSM, un tiers ou un service tel que celui-ci AWS CA privée peut générer le certificat.

   Chargez le certificat racine dans AWS Payment Cryptography à l'aide de la `importKey` commande avec KeyMaterialType of `RootCertificatePublicKey` et KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

   Pour les certificats intermédiaires, utilisez la `importKey` commande avec KeyMaterialType of `TrustedCertificatePublicKey` et KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Répétez cette procédure pour plusieurs certificats intermédiaires. Utilisez le dernier certificat importé `KeyArn` de la chaîne comme entrée pour les commandes d'importation suivantes.

   > **Note**
   >
   > N'importez pas le certificat Leaf. Fournissez-le directement lors de la commande d'importation.

5. Dérivez une clé unique à l'aide de l'ECDH sur le HSM Party U

   De nombreux HSM et systèmes associés prennent en charge l'établissement de clés à l'aide de l'ECDH. Spécifiez la clé publique de l'étape 1 comme clé publique et la clé de l'étape 3 comme clé privée. Pour les options autorisées, telles que les méthodes de dérivation, consultez le guide de l'API.

> **Note**
>
> Les paramètres de dérivation tels que le type de hachage doivent correspondre exactement des deux côtés. Dans le cas contraire, vous allez générer une clé différente.

6. Exporter la clé depuis le système source

   Enfin, exportez la clé que vous souhaitez transférer vers AWS Payment Cryptography à l'aide de TR-31 commandes standard. Spécifiez la clé dérivée de l'ECDH en tant que KBPK. La clé à exporter peut être n'importe quelle clé TDES ou AES soumise à des combinaisons TR-31 valides, à condition que la clé d'encapsulation soit au moins aussi puissante que la clé à exporter.

7. Clé d'importation d'appels

   Appelez l'`import-key`API avec un KeyMaterialType de`DiffieHellmanTr31KeyBlock`. Utilisez le KEYarn de la dernière autorité de certification importée à l'étape 3 pour`certificate-authority-public-key-identifier`, le matériel clé encapsulé de l'étape 4 pour `key-material` et le certificat feuille de l'étape 3 pour. `public-key-certificate` Incluez l'ARN de la clé privée indiqué à l'étape 1.

```
$ aws payment-cryptography import-key \
          --key-material='{
            "DiffieHellmanTr31KeyBlock": {
              "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-2:111122223333:key/swseahwtq2oj6zi5",
              "DerivationData": {
                "SharedInformation": "1234567890"
              },
              "DeriveKeyAlgorithm": "AES_256",
              "KeyDerivationFunction": "NIST_SP800",
              "KeyDerivationHashAlgorithm": "SHA_256",
              "PrivateKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv",
              "PublicKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN....",
              "WrappedKeyBlock":
 "D0112K1TB00E0000D603CCA8ACB71517906600FF8F0F195A38776A7190A0EF0024F088A5342DB98E2735084A7
            }
          }'
```

```
{
        "Key": {
          "CreateTimestamp": "2025-03-13T16:52:52.859000-04:00",
          "Enabled": true,
          "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
          "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
              "Decrypt": true,
              "DeriveKey": false,
              "Encrypt": true,
              "Generate": false,
              "NoRestrictions": false,
              "Sign": false,
              "Unwrap": true,
              "Verify": false,
              "Wrap": true
            },
            "KeyUsage": "TR31_K1_KEY_ENCRYPTION_KEY"
          },
          "KeyCheckValue": "CB94A2",
          "KeyCheckValueAlgorithm": "ANSI_X9_24",
          "KeyOrigin": "EXTERNAL",
          "KeyState": "CREATE_COMPLETE",
          "UsageStartTimestamp": "2025-03-13T16:52:52.859000-04:00"
        }
      }
```


8. Utiliser la clé importée pour les opérations cryptographiques ou les importations ultérieures

   Si la clé importée KeyUsage était TR31_K0_KEY_ENCRYPTION_KEY, vous pouvez utiliser cette clé pour les importations de clés suivantes en utilisant. TR-31 Pour les autres types de clés (tels que TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), vous pouvez utiliser la clé directement pour les opérations cryptographiques.

Importation de clés à l'aide de techniques asymétriques (RSA Unwrap)

Vue d'ensemble : La cryptographie des AWS paiements prend en charge le RSA wrap/unwrap pour l'échange de clés lorsque cela TR-34 n'est pas possible. Par exemple TR-34, cette technique utilise

la cryptographie asymétrique RSA pour chiffrer les clés symétriques à des fins d'échange. Cependant TR-34, contrairement à cette méthode, l'expéditeur ne signe pas la charge utile. De plus, cette technique d'encapsulation RSA ne préserve pas l'intégrité des métadonnées clés pendant le transfert car elle n'inclut pas les blocs clés.

> **Note**
>
> Vous pouvez utiliser RSA wrap pour importer ou exporter des TDES et AES-128 des clés.

1. Appelez la commande Initialiser l'importation

   Appelez get-parameters-for-import pour initialiser le processus d'importation avec un `KeyMaterialType` de`KEY_CRYPTOGRAM`. `RSA_2048`À utiliser `WrappingKeyAlgorithm` lors de l'échange de clés TDES. Utilisez `RSA_3072` ou `RSA_4096` lors de l'échange de TDES ou de AES-128 clés. Cette API génère une paire de clés pour les importations de clés, signe la clé à l'aide d'une racine de certificat et renvoie à la fois le certificat et la racine du certificat. Chiffrez la clé à exporter à l'aide de cette clé. Ces certificats sont de courte durée et ne sont destinés qu'à cette fin.

   ```
   $ aws payment-cryptography get-parameters-for-import \
       --key-material-type KEY_CRYPTOGRAM \
       --wrapping-key-algorithm RSA_4096
   ```

   ```
   {
     "ImportToken": "import-token-bwxli6ocftypneu5",
     "ParametersValidUntilTimestamp": 1698245002.065,
     "WrappingKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0....",
     "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0....",
     "WrappingKeyAlgorithm": "RSA_4096"
   }
   ```

2. Installer le certificat public sur le système source clé

   Avec de nombreux HSM, vous devez installer, charger ou approuver le certificat public (and/orsa racine) généré à l'étape 1 pour exporter les clés à l'aide de celui-ci.

3. Exporter la clé depuis le système source

De nombreux HSM et systèmes associés prennent en charge l'exportation de clés à l'aide de RSA Wrap. Spécifiez la clé publique de l'étape 1 en tant que certificat de chiffrement (`WrappingKeyCertificate`). Si vous avez besoin de la chaîne de confiance, utilisez `WrappingKeyCertificateChain` l'étape 1. Lorsque vous exportez la clé depuis votre HSM, spécifiez le format RSA, avec le mode de remplissage = PKCS #1 v2.2 OAEP (avec SHA 256 ou SHA 512).

4. Appel import-key

   Appelez l'import-keyAPI avec un `KeyMaterialType` de`KeyMaterial`. Vous avez besoin du `ImportToken` formulaire de l'étape 1 et du `key-material` (matériel clé emballé) de l'étape 3. Fournissez les paramètres clés (tels que l'utilisation des clés) car RSA Wrap n'utilise pas de blocs clés.

```
$ cat import-key-cryptogram.json
```

```
{
  "KeyMaterial": {
    "KeyCryptogram": {
      "Exportable": true,
      "ImportToken": "import-token-bwxli6ocftypneu5",
      "KeyAttributes": {
        "KeyAlgorithm": "AES_128",
        "KeyClass": "SYMMETRIC_KEY",
        "KeyModesOfUse": {
          "Decrypt": true,
          "DeriveKey": false,
          "Encrypt": true,
          "Generate": false,
          "NoRestrictions": false,
          "Sign": false,
          "Unwrap": true,
          "Verify": false,
          "Wrap": true
        },
        "KeyUsage": "TR31_K0_KEY_ENCRYPTION_KEY"
      },
      "WrappedKeyCryptogram": "18874746731....",
      "WrappingSpec": "RSA_OAEP_SHA_256"
    }
  }
```

```
}
```

```
$ aws payment-cryptography import-key --cli-input-json file://import-key-
cryptogram.json
```

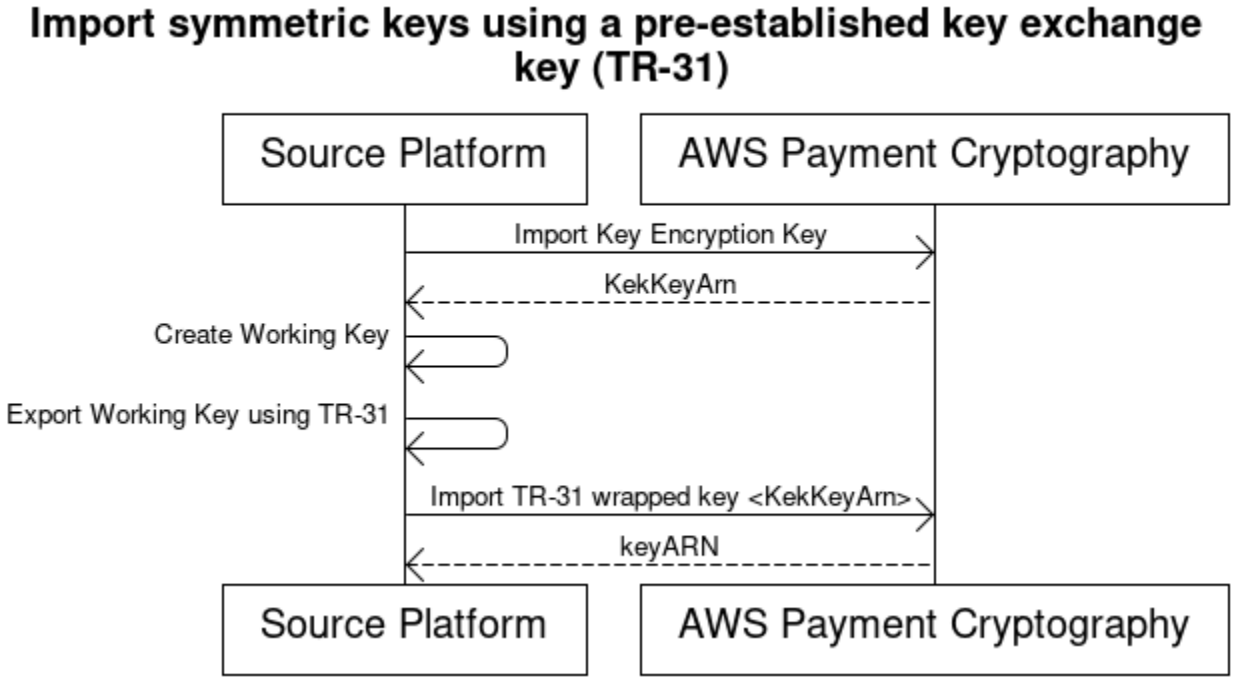
```
{
  "Key": {
    "KeyOrigin": "EXTERNAL",
    "Exportable": true,
    "KeyCheckValue": "DA1ACF",
    "UsageStartTimestamp": 1697643478.92,
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
    "CreateTimestamp": 1697643478.92,
    "KeyState": "CREATE_COMPLETE",
    "KeyAttributes": {
      "KeyAlgorithm": "AES_128",
      "KeyModesOfUse": {
        "Encrypt": true,
        "Unwrap": true,
        "Verify": false,
        "DeriveKey": false,
        "Decrypt": true,
        "NoRestrictions": false,
        "Sign": false,
        "Wrap": true,
        "Generate": false
      },
      "KeyUsage": "TR31_K0_KEY_ENCRYPTION_KEY",
      "KeyClass": "SYMMETRIC_KEY"
    },
    "KeyCheckValueAlgorithm": "CMAC"
  }
}
```

5. Utiliser la clé importée pour les opérations cryptographiques ou les importations ultérieures

   Si l'importation `KeyUsage` était `TR31_K0_KEY_ENCRYPTION_KEY` ou`TR31_K1_KEY_BLOCK_PROTECTION_KEY`, vous pouvez utiliser cette clé pour les importations de clés suivantes en utilisant TR-31. Si le type de clé était un autre type (tel

queTR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), vous pouvez utiliser la clé directement pour les opérations cryptographiques.

Importer des clés symétriques à l'aide d'une clé d'échange de clés préétablie () TR-31



Lorsqu'ils échangent plusieurs clés ou prennent en charge la rotation des clés, les partenaires échangent généralement d'abord une clé de chiffrement initiale (KEK). Vous pouvez échanger des KEK avec la cryptographie des AWS paiements, en utilisant des techniques telles que TR-34ouÉchange de clés physiques.

Après avoir créé un KEK, vous pouvez l'utiliser pour transporter les clés suivantes (y compris d'autres KEK). AWS La cryptographie des paiements prend en charge cet échange de clés à l'aide de la norme ANSI TR-31, qui est largement utilisée et prise en charge par les fournisseurs de HSM.

1. Clé d'importation Clé de chiffrement (KEK)

   Assurez-vous que vous avez déjà importé votre KEK et que le KeyArn (ou KeyAlias) est disponible.

2. Créer une clé sur la plateforme source

   Si la clé n'existe pas, créez-la sur la plateforme source. Vous pouvez également créer la clé sur AWS Payment Cryptography et utiliser la export commande.

3. Exporter la clé depuis la plateforme source

   Lors de l'exportation, spécifiez le format d'exportation sous la forme TR-31. La plateforme source vous demandera la clé à exporter et la clé de chiffrement à utiliser.

4. Importation dans la cryptographie des AWS paiements

   Lorsque vous appelez la import-key commande, utilisez le KeyArn (ou alias) de votre clé de chiffrement pour`WrappingKeyIdentifier`. Utilisez le résultat de la plate-forme source pour`WrappedKeyBlock`.

Example

```
$ aws payment-cryptography import-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza", \
    "WrappedKeyBlock":
 "D0112B0AX00E00002E0A3D58252CB67564853373D1EBCC1E23B2ADE7B15E967CC27B85D5999EF58E11662991F
 \
    }'
```

```
{
  "Key": {
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
    "KeyAttributes": {
      "KeyUsage": "TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY",
      "KeyClass": "SYMMETRIC_KEY",
      "KeyAlgorithm": "AES_128",
      "KeyModesOfUse": {
        "Encrypt": true,
        "Decrypt": true,
        "Wrap": true,
        "Unwrap": true,
        "Generate": false,
        "Sign": false,
        "Verify": false,
        "DeriveKey": false,
        "NoRestrictions": false
      }
    },
    "KeyCheckValue": "0A3674",
    "KeyCheckValueAlgorithm": "CMAC",
    "Enabled": true,
    "Exportable": true,
    "KeyState": "CREATE_COMPLETE",
    "KeyOrigin": "EXTERNAL",
    "CreateTimestamp": "2023-06-02T07:38:14.913000-07:00",
    "UsageStartTimestamp": "2023-06-02T07:38:14.857000-07:00"
  }
}
```

Importation de clés publiques asymétriques (RSA, ECC)

Tous les certificats importés doivent être au moins aussi solides que le certificat émetteur (prédécesseur) de la chaîne. Cela signifie qu'une autorité de certification RSA_2048 ne peut être utilisée que pour protéger un certificat feuille RSA_2048 et qu'un certificat ECC doit être protégé par un autre certificat ECC de force équivalente. Un certificat ECC P384 ne peut être délivré que par une autorité de certification P384 ou P521. Tous les certificats ne doivent pas être expirés au moment de l'importation.

Importation de clés publiques RSA

AWS La cryptographie des paiements prend en charge l'importation de clés RSA publiques sous forme X.509 de certificats. Pour importer un certificat, importez d'abord son certificat racine. Tous les certificats ne doivent pas être expirés au moment de l'importation. Le certificat doit être au format PEM et encodé en base64.

1. Importer le certificat racine dans la cryptographie des AWS paiements

   Utilisez la commande suivante pour importer le certificat racine :

Example

2. Importer un certificat à clé publique dans la cryptographie des AWS paiements

   Vous pouvez désormais importer une clé publique. Comme TR-34 l'ECDH repose sur la transmission du certificat Leaf au moment de l'exécution, cette option n'est utilisée que lors du chiffrement de données à l'aide d'une clé publique provenant d'un autre système. KeyUsage sera défini sur TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION.

Example

```
$ aws payment-cryptography import-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza", \
    "WrappedKeyBlock":
 "D0112B0AX00E00002E0A3D58252CB67564853373D1EBCC1E23B2ADE7B15E967CC27B85D5999EF58E11662991F
 \
    }'
```

```
{
  "Key": {
    "CreateTimestamp": "2023-08-08T18:55:46.815000+00:00",
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/4kd6xud22e64wcbk",
    "KeyAttributes": {
      "KeyAlgorithm": "RSA_4096",
      "KeyClass": "PUBLIC_KEY",
      "KeyModesOfUse": {
        "Decrypt": false,
        "DeriveKey": false,
        "Encrypt": false,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": false,
        "Verify": true,
        "Wrap": false
      },
      "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
    },
    "KeyOrigin": "EXTERNAL",
    "KeyState": "CREATE_COMPLETE",
    "UsageStartTimestamp": "2023-08-08T18:55:46.815000+00:00"
  }
}
```

Importation de clés publiques ECC

AWS La cryptographie des paiements prend en charge l'importation de clés ECC publiques sous forme X.509 de certificats. Pour importer un certificat, importez d'abord son certificat CA racine et tous les certificats intermédiaires. Tous les certificats ne doivent pas être expirés au moment de l'importation. Le certificat doit être au format PEM et encodé en base64.

1. Importer le certificat racine ECC dans la cryptographie des AWS paiements

   Utilisez la commande suivante pour importer le certificat racine :

Example

2. Importer un certificat intermédiaire dans la cryptographie des AWS paiements

   Utilisez la commande suivante pour importer un certificat intermédiaire :

Example

3. Importer un certificat à clé publique (Leaf) dans la cryptographie des AWS paiements

   Bien que vous puissiez importer un certificat ECC en feuille, il n'existe actuellement aucune fonction définie dans AWS Payment Cryptography en dehors du stockage. En effet, lors de l'utilisation des fonctions ECDH, le certificat feuille est transmis au moment de l'exécution.

Clés d'exportation

Table des matières

Exporter des clés symétriques

> **Important**
>
> Assurez-vous de disposer de la dernière version de AWS CLI avant de commencer. Pour effectuer une mise à niveau, reportez-vous à la section Installation du AWS CLI.

Exporter des clés à l'aide de techniques asymétriques () TR-34

TR-34 utilise le chiffrement asymétrique RSA pour chiffrer et signer des clés symétriques à des fins d'échange. Le cryptage protège la confidentialité, tandis que la signature garantit l'intégrité. Lorsque vous exportez des clés, AWS Payment Cryptography agit en tant qu'hôte de distribution des clés (KDH) et votre système cible devient le dispositif de réception des clés (KRD).

> **Note**
>
> Si votre HSM prend en charge TR-34 l'exportation mais pas l' TR-34 importation, nous vous recommandons d'abord d'établir un KEK partagé entre votre HSM et AWS Payment Cryptography à l'aide de. TR-34 Vous pouvez ensuite l'utiliser TR-31 pour transférer vos clés restantes.

1. Initialisation du processus d'exportation

   Exécutez get-parameters-for-export pour générer une paire de clés pour les exportations de clés. Nous utilisons cette paire de clés pour signer la TR-34 charge utile. En TR-34 termes terminologiques, il s'agit du certificat de signature du KDH. Les certificats sont de courte durée et ne sont valables que pour la durée spécifiée dans`ParametersValidUntilTimestamp`.

   > **Note**
   >
   > Tous les certificats sont encodés en base64.

   Example

   ```
   $ aws payment-cryptography get-parameters-for-export \
       --signing-key-algorithm RSA_2048 \
       --key-material-type TR34_KEY_BLOCK
   ```

   ```
   {
     "SigningKeyCertificate":
    "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJ...",
     "SigningKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS....",
     "SigningKeyAlgorithm": "RSA_2048",
     "ExportToken": "export-token-au7pvkbsq4mbup6i",
     "ParametersValidUntilTimestamp": "2023-06-13T15:40:24.036000-07:00"
   }
   ```

2. Importez le certificat AWS de cryptographie des paiements dans votre système de réception

   Importez la chaîne de certificats de l'étape 1 dans votre système de réception.

3. Configurez les certificats de votre système de réception

Pour protéger la charge utile transmise, l'expéditeur (KDH) la chiffre. Votre système de réception (généralement votre HSM ou le HSM de votre partenaire) doit générer une clé publique et créer un certificat de clé X.509 publique. Vous pouvez l'utiliser AWS CA privée pour générer des certificats, mais vous pouvez utiliser n'importe quelle autorité de certification.

Une fois le certificat obtenu, importez le certificat racine dans AWS Payment Cryptography à l'aide de la ImportKey commande. Définissez `KeyMaterialType` sur `RootCertificatePublicKey` et `KeyUsageType` sur `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Nous l'utilisons `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` comme clé `KeyUsageType` car il s'agit de la clé racine qui signe le certificat feuille. Il n'est pas nécessaire d'importer des certificats Leaf dans AWS Payment Cryptography : vous pouvez les transmettre en ligne.

> **Note**
>
> Si vous avez déjà importé le certificat racine, ignorez cette étape. Pour les certificats intermédiaires, utilisez`TrustedCertificatePublicKey`.

4. Exportez votre clé

   Appelez l'ExportKeyAPI avec `KeyMaterialType` set to`TR34_KEY_BLOCK`. Vous devez fournir :

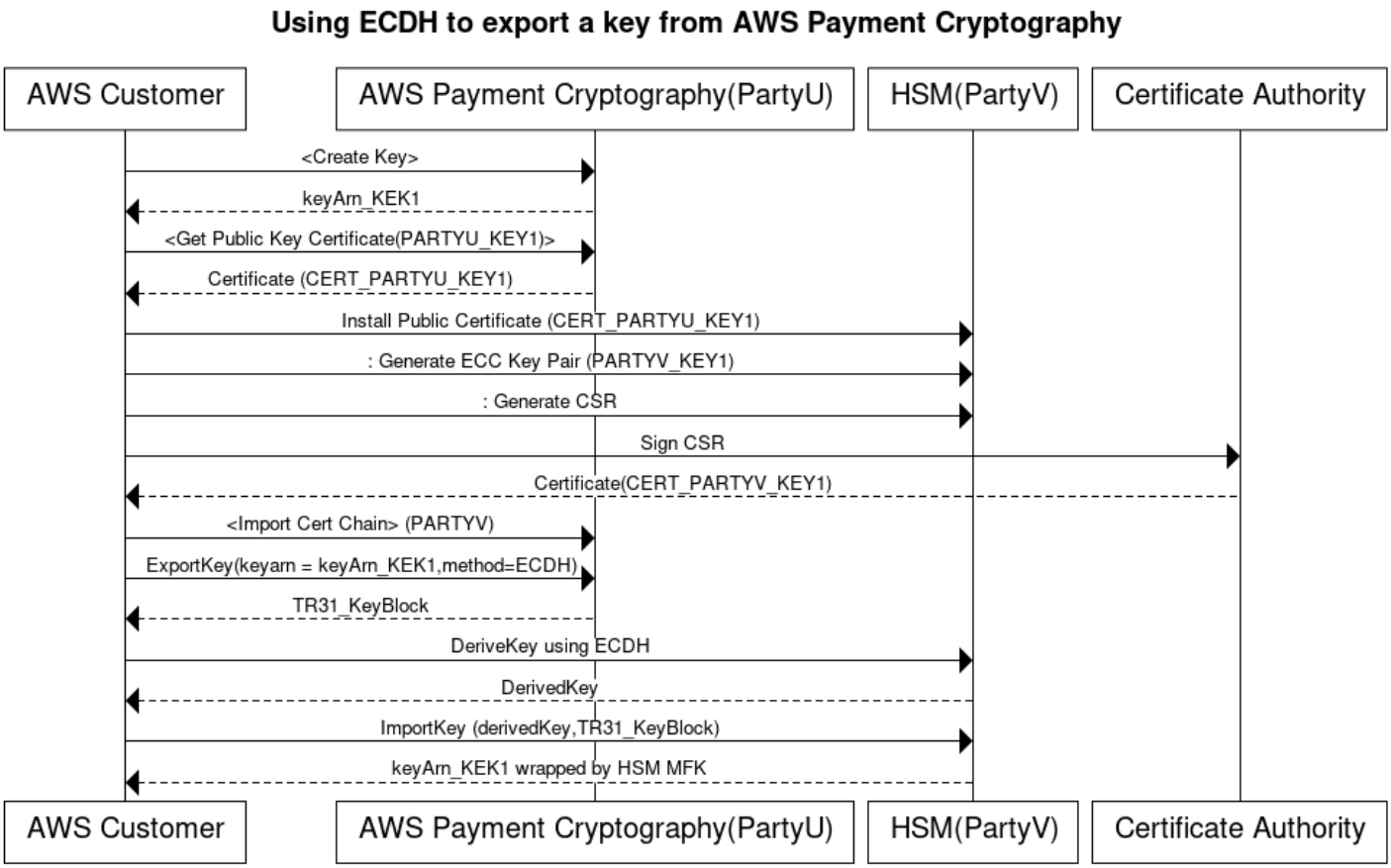
   - Le KeyArn de l'autorité de certification racine de l'étape 3 est `CertificateAuthorityPublicKeyIdentifier`
   - Le certificat Leaf de l'étape 3 en tant que `WrappingKeyCertificate`
   - Le KeyArn (ou alias) de la clé que vous souhaitez exporter en tant que `--export-key-identifier`
   - Le jeton d'exportation de l'étape 1

Example

```
$ aws payment-cryptography export-key \
    --export-key-identifier "example-export-key" \
    --key-material '{"Tr34KeyBlock": { \
    "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/4kd6xud22e64wcbk", \
    "ExportToken": "export-token-au7pvkbsq4mbup6i", \
    "KeyBlockFormat": "X9_TR34_2012", \
    "WrappingKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFXZ0F3SUJBZ0lSQ..."} \
    }'
```

```
{
  "WrappedKey": {
    "KeyMaterial": "308205A106092A864886F70D010702A08205923082058...",
    "WrappedKeyMaterialFormat": "TR34_KEY_BLOCK"
  }
}
```

Exporter des clés à l'aide de techniques asymétriques (ECDH)



Elliptic Curve Diffie-Hellman (ECDH) utilise la cryptographie asymétrique ECC pour établir une clé partagée entre deux parties sans nécessiter de clés pré-échangées. Les clés ECDH étant éphémères, AWS Payment Cryptography ne les stocke pas. Dans ce processus, une seule fois KBPK/KEKest dérivée à l'aide de l'ECDH. Cette clé dérivée est immédiatement utilisée pour encapsuler la clé que vous souhaitez transférer, qui peut être un autre KBPK, un BDK, une clé IPEK ou d'autres types de clés.

Lors de l'exportation, la cryptographie des AWS paiements est appelée partie U (initiateur) et le système de réception est appelé partie V (répondeur).

> **Note**
>
> L'ECDH peut être utilisé pour échanger n'importe quel type de clé symétrique, mais c'est la seule approche qui peut être utilisée pour transférer des AES-256 clés si aucun KEK n'est déjà établi.

1. Générer une paire de clés ECC

   Appelez `create-key` pour créer une paire de clés ECC pour ce processus. Cette API génère une paire de clés pour les importations ou les exportations de clés. Lors de la création, spécifiez le type de clés pouvant être dérivées à l'aide de cette clé ECC. Lorsque vous utilisez l'ECDH pour échanger (encapsuler) d'autres clés, utilisez une valeur de`TR31_K1_KEY_BLOCK_PROTECTION_KEY`.

   > **Note**
   >
   > Bien que l'ECDH de bas niveau génère une clé dérivée qui peut être utilisée à n'importe quelle fin, la cryptographie des AWS paiements limite la réutilisation accidentelle d'une clé à des fins multiples en autorisant l'utilisation d'une clé uniquement pour un seul type de clé dérivée.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=ECC_NIST_P256,KeyUsage=TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT,KeyClass=ASYM
 --derive-key-usage "TR31_K1_KEY_BLOCK_PROTECTION_KEY"
```

```
{
        "Key": {
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
wc3rjsssguhxtilv",
            "KeyAttributes": {
                "KeyUsage": "TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT",
                "KeyClass": "ASYMMETRIC_KEY_PAIR",
                "KeyAlgorithm": "ECC_NIST_P256",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
```

```
            "KeyCheckValue": "2432827F",
            "KeyCheckValueAlgorithm": "CMAC",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2025-03-28T22:03:41.087000-07:00",
            "UsageStartTimestamp": "2025-03-28T22:03:41.068000-07:00"
        }
    }
```

2. Obtenir un certificat de clé publique

   Appelez `get-public-key-certificate` pour recevoir la clé publique sous forme de X.509 certificat signé par l'autorité de certification de votre compte, spécifique à la cryptographie des AWS paiements dans une région spécifique.

   Example

```
$ aws payment-cryptography get-public-key-certificate \
          --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv
```

```
{
      "KeyCertificate": "LS0tLS1CRUdJTi...",
      "KeyCertificateChain": "LS0tLS1CRUdJT..."
    }
```

3. Installer un certificat public sur le système de contrepartie (Partie V)

   Pour de nombreux HSM, vous devez installer, charger ou approuver le certificat public généré à l'étape 1 pour établir les clés. Cela peut inclure l'ensemble de la chaîne de certificats ou uniquement le certificat racine, selon le HSM. Consultez la documentation de votre HSM pour obtenir des instructions spécifiques.

4. Générez une paire de clés ECC sur le système source et fournissez une chaîne de certificats à AWS Payment Cryptography

   Dans l'ECDH, chaque partie génère une paire de clés et s'accorde sur une clé commune. Pour que AWS Payment Cryptography puisse dériver la clé, elle a besoin de la clé publique de la contrepartie au format clé X.509 publique.

Lorsque vous transférez des clés depuis un HSM, créez une paire de clés sur ce HSM. Pour les HSM qui prennent en charge les blocs de touches, l'en-tête de clé ressemblera à`D0144K3EX00E0000`. Lors de la création du certificat, vous générez généralement un CSR sur le HSM, puis le HSM, un tiers ou un service tel que celui-ci AWS CA privée peut générer le certificat.

Chargez le certificat racine dans AWS Payment Cryptography à l'aide de la `importKey` commande avec KeyMaterialType of `RootCertificatePublicKey` et KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Pour les certificats intermédiaires, utilisez la `importKey` commande avec KeyMaterialType of `TrustedCertificatePublicKey` et KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Répétez cette procédure pour plusieurs certificats intermédiaires. Utilisez le dernier certificat importé `KeyArn` de la chaîne comme entrée pour les commandes d'exportation suivantes.

> **Note**
>
> N'importez pas le certificat Leaf. Fournissez-le directement lors de la commande d'exportation.

5. Dérivez la clé et exportez la clé à partir de la cryptographie des AWS paiements

Lors de l'exportation, le service déduit une clé à l'aide de l'ECDH, puis l'utilise immédiatement comme KBPK pour encapsuler la clé à exporter. TR-31 La clé à exporter peut être n'importe quelle clé TDES ou AES soumise à des combinaisons TR-31 valides, à condition que la clé d'encapsulation soit au moins aussi puissante que la clé à exporter.

```
$ aws payment-cryptography export-key \
            --export-key-identifier arn:aws:payment-cryptography:us-
west-2:529027455495:key/e3a65davqhbpjm4h \
            --key-material='{
              "DiffieHellmanTr31KeyBlock": {
                "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-2:111122223333:key/swseahwtq2oj6zi5",
                "DerivationData": {
                  "SharedInformation": "ADEF567890"
                },
                "DeriveKeyAlgorithm": "AES_256",
```

```
                "KeyDerivationFunction": "NIST_SP800",
                "KeyDerivationHashAlgorithm": "SHA_256",
                "PrivateKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv",
                "PublicKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FUR..."
              }
            }'
```

```
{
            "WrappedKey": {
                "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK",
                "KeyMaterial":
 "D0112K1TB00E00007012724C0FAAF64DA50E2FF4F9A94DF50441143294E0E995DB2171554223EAA56D078C4CF
                "KeyCheckValue": "E421AD",
                "KeyCheckValueAlgorithm": "ANSI_X9_24"
            }
        }
```

6. Dérivez une clé unique à l'aide de l'ECDH sur le HSM Party V

   De nombreux HSM et systèmes associés prennent en charge l'établissement de clés à l'aide de l'ECDH. Spécifiez la clé publique de l'étape 1 comme clé publique et la clé de l'étape 3 comme clé privée. Pour les options autorisées, telles que les méthodes de dérivation, consultez le guide de l'API.

   > **Note**
   >
   > Les paramètres de dérivation tels que le type de hachage doivent correspondre exactement des deux côtés. Dans le cas contraire, vous allez générer une clé différente.

7. Importer la clé vers le système cible

   Enfin, importez la clé depuis AWS Payment Cryptography à l'aide de TR-31 commandes standard. Spécifiez la clé dérivée de l'ECDH en tant que KBPK et utilisez le bloc de TR-31 clé précédemment exporté depuis AWS Payment Cryptography.

Exporter des clés à l'aide de techniques asymétriques (RSA Wrap)

Lorsque TR-34 ce n'est pas disponible, vous pouvez utiliser RSA wrap/unwrap pour l'échange de clés. Par exemple TR-34, cette méthode utilise la cryptographie asymétrique RSA pour chiffrer les clés symétriques. Toutefois, l'enveloppe RSA n'inclut pas :

- Signature de la charge utile par l'expéditeur
- Blocs clés qui préservent l'intégrité des métadonnées clés pendant le transport

> **Note**
>
> Vous pouvez utiliser RSA wrap pour exporter des TDES et AES-128 des clés.

1. Créez une clé et un certificat RSA sur votre système de réception

   Créez ou identifiez une clé RSA pour recevoir la clé encapsulée. Nous exigeons que les clés soient au format de X.509 certificat. Assurez-vous que le certificat est signé par un certificat racine que vous pouvez importer dans AWS Payment Cryptography.
2. Importer le certificat public racine dans AWS Payment Cryptography

   À utiliser import-key avec l'`--key-material`option d'importation du certificat

```
$ aws payment-cryptography import-key \
    --key-material='{"RootCertificatePublicKey": { \
    "KeyAttributes": { \
    "KeyAlgorithm": "RSA_4096", \
    "KeyClass": "PUBLIC_KEY", \
    "KeyModesOfUse": {"Verify": true}, \
    "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"}, \
    "PublicKeyCertificate": "LS0tLS1CRUdJTiBDRV..."} \
    }'
```

```
{
  "Key": {
    "CreateTimestamp": "2023-09-14T10:50:32.365000-07:00",
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
nsq2i3mbg6sn775f",
    "KeyAttributes": {
```

```
      "KeyAlgorithm": "RSA_4096",
      "KeyClass": "PUBLIC_KEY",
      "KeyModesOfUse": {
        "Decrypt": false,
        "DeriveKey": false,
        "Encrypt": false,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": false,
        "Verify": true,
        "Wrap": false
      },
      "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
    },
    "KeyOrigin": "EXTERNAL",
    "KeyState": "CREATE_COMPLETE",
    "UsageStartTimestamp": "2023-09-14T10:50:32.365000-07:00"
  }
}
```

3. Exportez votre clé

   Demandez à AWS Payment Cryptography d'exporter votre clé à l'aide de votre certificat Leaf. Vous devez spécifier :

   - L'ARN du certificat racine que vous avez importé à l'étape 2
   - Le certificat Leaf pour l'exportation
   - La clé symétrique à exporter

   La sortie est une version encapsulée (cryptée) binaire codée en hexadécimal de votre clé symétrique.

Example Exemple — Exportation d'une clé

```
$ cat export-key.json
```

```
{
  "ExportKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
  "KeyMaterial": {
    "KeyCryptogram": {
      "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/zabouwe3574jysdl",
      "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDEXAMPLE...",
      "WrappingSpec": "RSA_OAEP_SHA_256"
    }
  }
}
```

```
$ aws payment-cryptography export-key \
    --cli-input-json file://export-key.json
```

```
{
  "WrappedKey": {
    "KeyMaterial":
 "18874746731E9E1C4562E4116D1C2477063FCB08454D757D81854AEAEE0A52B1F9D303FA29C02DC82AE778535
    "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM"
  }
}
```

4. Importez la clé dans votre système de réception

   De nombreux HSM et systèmes associés prennent en charge l'importation de clés à l'aide de RSA unwrap (y compris la cryptographie des AWS paiements). Lors de l'importation, spécifiez :

   - La clé publique de l'étape 1 en tant que certificat de chiffrement
   - Le format en tant que RSA
   - Mode de rembourrage en tant que PKCS #1 v2.2 OAEP (avec SHA 256)

> **Note**
>
> Nous produisons la clé encapsulée au format HexBinary. Vous devrez peut-être convertir le format si votre système nécessite une représentation binaire différente, telle que base64.

Exporter des clés symétriques à l'aide d'une clé d'échange de clés préétablie () TR-31

Lorsqu'ils échangent plusieurs clés ou prennent en charge la rotation des clés, les partenaires échangent généralement d'abord une clé de chiffrement initiale (KEK). Vous pouvez échanger des KEK avec la cryptographie des AWS paiements, en utilisant des techniques telles que TR-34ouÉchange de clés physiques. Après avoir créé un KEK, vous pouvez l'utiliser pour transporter les clés suivantes, y compris d'autres KEK. Nous prenons en charge cet échange de clés à l'aide de la norme ANSI TR-31, qui est largement prise en charge par les fournisseurs de HSM.

1. Configurez votre clé de chiffrement (KEK)

   Assurez-vous que vous avez déjà échangé votre KEK et que le KeyArn (ou KeyAlias) est disponible.

2. Créez votre clé sur la cryptographie des AWS paiements

   Créez votre clé si elle n'existe pas déjà. Vous pouvez également créer la clé sur votre autre système et utiliser la commande d'importation.

3. Exportez votre clé depuis AWS Payment Cryptography

   Lorsque vous exportez au TR-31 format, spécifiez la clé que vous souhaitez exporter et la clé d'encapsulage à utiliser.

Example Exemple — Exportation d'une clé à l'aide du bloc de touches TR31

```
$ aws payment-cryptography export-key \
    --key-material='{"Tr31KeyBlock": \
    { "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza" }}' \
    --export-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp
```

```
{
  "WrappedKey": {
    "KeyCheckValue": "73C263",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "KeyMaterial":
 "D0144K0AB00E0000A24D3ACF3005F30A6E31D533E07F2E1B17A2A003B338B1E79E5B3AD4FBF7850FACF9A3784
    "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK"
  }
}
```

4. Importez la clé dans votre système

   Utilisez l'implémentation de la clé d'importation de votre système pour importer la clé.

Exporter les clés initiales DUKPT () IPEK/IK

Lorsque vous utilisez DUKPT, vous pouvez générer une clé de dérivation de base (BDK) unique pour un parc de terminaux. Les terminaux n'ont pas d'accès direct au BDK. Au lieu de cela, chaque terminal reçoit une clé de terminal initiale unique, connue sous le nom d'IPEK ou Initial Key (IK). Chaque IPEK est dérivé du BDK à l'aide d'un numéro de série de clé (KSN) unique.

La structure du KSN varie en fonction du type de chiffrement :

- Pour TDES : le KSN à 10 octets inclut :
  - 24 bits pour l'ID du jeu de clés
  - 19 bits pour l'ID du terminal
  - 21 bits pour le compteur de transactions
- Pour AES : le KSN à 12 octets inclut :
  - 32 bits pour l'ID BDK

- 32 bits pour l'identifiant de dérivation (ID)
- 32 bits pour le compteur de transactions

Nous fournissons un mécanisme pour générer et exporter ces clés initiales. Vous pouvez exporter les clés générées à l'aide des TR-31 méthodes d'encapsulation RSA ou RSA. TR-34 Notez que les clés IPEK ne sont pas conservées et ne peuvent pas être utilisées pour des opérations ultérieures sur AWS la cryptographie des paiements.

Nous n'imposons pas la division entre les deux premières parties du KSN. Si vous souhaitez enregistrer l'identifiant de dérivation avec le BDK, vous pouvez utiliser AWS des balises.

> **Note**
>
> La partie compteur du KSN (32 bits pour AES DUKPT) n'est pas utilisée pour la dérivation. IPEK/IK Par exemple, les entrées 12345678901234560001 et 12345678901234569999 généreront le même IPEK.

```
$ aws payment-cryptography export-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza"}} ' \
    --export-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --export-attributes 'ExportDukptInitialKey={KeySerialNumber=12345678901234560001}'
```

```
{
"WrappedKey": {
    "KeyCheckValue": "73C263",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "KeyMaterial":
 "B0096B1TX00S000038A8A06588B9011F0D5EEF1CCAECFA6962647A89195B7A98BDA65DDE7C57FEA507559AF2A5D60
    "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK"
}
}
```

Spécifier les en-têtes des blocs-clés pour l'exportation

Vous pouvez modifier ou ajouter des informations sur les blocs clés lors de l'exportation au format ASC TR-31 ou TR-34 dans des formats. Le tableau suivant décrit le format du bloc TR-31 clé et les éléments que vous pouvez modifier lors de l'exportation.

Attribut du bloc clé	Objectif	Pouvez-vous modifier lors de l'exportation ?	Remarques
ID de version	Définit la méthode utilisée pour protéger le matériau clé. La norme inclut : • Versions A et C (variante clé - obsolète) • Version B (dérivation à l'aide du TDES) • Version D (dérivation des clés à l'aide d'AES)	Non	Nous utilisons la version B pour les clés d'encapsulation TDES et la version D pour les clés d'encapsulation AES. Nous prenons en charge les versions A et C uniquement pour les opérations d'importation.
Longueur du bloc clé	Spécifie la longueur du message restant	Non	Nous calculons cette valeur automatiquement. La longueur peut sembler incorrecte avant de déchiffrer la charge utile, car nous pouvons ajouter un rembourrage des touches conformément à la spécification.

Attribut du bloc clé	Objectif	Pouvez-vous modifier lors de l'exportation ?	Remarques
Utilisation de la clé	Définit les objectifs autorisés pour la clé, tels que : • C0 (Vérification de la carte) • B0 (clé de dérivation de base)	Non	
Algorithm	Spécifie l'algorithme de la clé sous-jacente. Nous soutenons : • T (MARÉES) • H (HMAC) • UN (AS)	Non	Nous exportons cette valeur telle quelle.
Utilisation de la clé	Définit les opérations autorisées, telles que : • Générer et vérifier (C) • Encrypt/Decrypt/Wrap/Unwrap (B)	Oui*	
Version clé	Indique le numéro de version de la clé replacement/rotation. La valeur par défaut est 00 si elle n'est pas spécifiée.	Oui - Peut être ajouté	

Attribut du bloc clé	Objectif	Pouvez-vous modifier lors de l'exportation ?	Remarques
Exportabilité clé	Contrôle si la clé peut être exportée : • N - Aucune exportabilité • E - Exporter selon X9.24 (blocs clés) • S - Exporter sous des formats de bloc clé ou de bloc non clé	Oui*	
Blocs clés facultatifs	Oui - Peut être ajouté	Les blocs de clé facultatifs sont des name/value paires liées cryptographiquement à la clé. Par exemple, KeySet ID pour les clés DUKPT. Nous calculons automatiquement le nombre de blocs, la longueur de chaque bloc et le bloc de remplissage (PB) en fonction de votre name/value paire saisie.	

*Lorsque vous modifiez des valeurs, votre nouvelle valeur doit être plus restrictive que la valeur actuelle dans AWS Payment Cryptography. Par exemple :

- Si le mode d'utilisation des clés actuel est Generate=True, Verify=True, vous pouvez le remplacer par Generate=True, Verify=False
- Si la clé est déjà définie sur « non exportable », vous ne pouvez pas la modifier en « exportable »

Lorsque vous exportez des clés, nous appliquons automatiquement les valeurs actuelles de la clé exportée. Toutefois, vous souhaiterez peut-être modifier ou ajouter ces valeurs avant de les envoyer au système récepteur. Voici quelques scénarios courants :

- Lorsque vous exportez une clé vers un terminal de paiement, définissez son exportabilité sur, `Not Exportable` car les terminaux n'importent généralement que des clés et ne devraient pas les exporter.
- Lorsque vous devez transmettre les métadonnées de clé associées au système récepteur, utilisez des en-têtes TR-31 facultatifs pour lier cryptographiquement les métadonnées à la clé au lieu de créer une charge utile personnalisée.
- Définissez la version clé à l'aide du `KeyVersion` champ pour suivre la rotation des clés.

TR-31/X9.143 définit les en-têtes courants, mais vous pouvez utiliser d'autres en-têtes tant qu'ils respectent les paramètres de cryptographie des AWS paiements et que votre système de réception peut les accepter. Pour plus d'informations sur les en-têtes de blocs de touches lors de l'exportation, consultez la section En-têtes de blocs de touches du guide de l'API.

Voici un exemple d'exportation d'une clé BDK (par exemple, vers un KIF) avec les spécifications suivantes :

- Version clé : 02
- KeyExportability: NON EXPORTABLE
- KeySetID : 00ABCDEFAB (00 indique la clé TDES, ABCDEFABCD est la clé initiale)

Comme nous ne spécifiez pas les modes d'utilisation des clés, cette clé hérite du mode d'utilisation de arn:aws:payment-cryptography:us-east- 2:111122223333 : (= true). key/5rplquuwozodpwsp DeriveKey

> **Note**
>
> Même si vous définissez l'exportabilité sur Non exportable dans cet exemple, le KIF peut toujours :

- Clés de dérivation telles que IPEK/IKcelles utilisées dans DUKPT
- Exportez ces clés dérivées pour les installer sur les appareils

Cela est spécifiquement autorisé par les normes.

```
$ aws payment-cryptography export-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza", \
    "KeyBlockHeaders": { \
    "KeyModesOfUse": { \
    "Derive": true}, \
    "KeyExportability": "NON_EXPORTABLE", \
    "KeyVersion": "02", \
    "OptionalBlocks": { \
    "BI": "00ABCDEFABCD"}}} \
    }' \
    --export-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp
```

```
{
"WrappedKey": {
    "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK",
    "KeyMaterial": "EXAMPLE_KEY_MATERIAL_TR31",
    "KeyCheckValue": "A4C9B3",
    "KeyCheckValueAlgorithm": "ANSI_X9_24"
    }
}
```

En-têtes communs

X9.143 définit certains en-têtes pour les cas d'utilisation courants. À l'exception de l'en-tête HM (HMAC Hash), AWS Payment Cryptography n'analyse ni n'utilise ces en-têtes.

Nom de l'en-tête	Objectif	Validation typique	Remarques
BI	Identifiant de clé de dérivation de base pour DUKPT	2 caractères hexadécimaux (00 pour TDES, 11 pour AES) puis 10 caractères hexadécimaux pour TDES KSI ou 8 caractères hexadécimaux pour BDK ID (AES DUKPT).	Contient le (identifiant BDK, pour AES DUKPT) ou l'identifiant du jeu de clés (KSI, pour TDES DUKPT). Peut être utilisé lors de l'échange du BDK ID ou du KSI, mais il n'est pas nécessaire d'échanger les autres données contenues dans les blocs IK et KS. Généralement, le BI est utilisé lors de la transmission vers un KIF, tandis que l'IK ou le KS sont utilisés lors de l'injection dans le terminal lui-même.
HM	Spécifie le type de hachage pour les opérations HMAC	• 10 — SHA-1 • 20 — SHA-224 • 21 — SHA-256 • 22 — SHA-384 • 23 — SHA-512 • 24 — SHA-512/224 • 25 — SHA-512/256 • 30 — SHA3-224 • 31 — SHA3-256 • 32 — SHA3-384 • 33 — SHA3-512	Le service remplit automatiquement ce champ lors de l'exportation et l'analysera lors de l'importation. Les types de hachage non pris en charge par le service, tels que SHAKE128, peuvent être importés mais peuvent ne pas être utilisables pour les

Nom de l'en-tête	Objectif	Validation typique	Remarques
		• 40 — SHAKE128 • 41 — SHAKE256	fonctions cryptographiques.
cinématique inverse	Numéro de série de la clé initiale pour AES DUKPT	16 caractères hexadécimaux	Cette valeur est utilisée pour instancier l'utilisation de la clé DUKPT initiale sur le périphérique récepteur et identifie la clé initiale dérivée d'un BDK. Ce champ contient généralement les données de dérivation mais aucun compteur. Utilisez KS pour TDES DUKPT.
KS	Numéro de série de la clé initiale pour TDES DUKPT	20 caractères hexadécimaux	Cette valeur est utilisée pour instancier l'utilisation de la clé DUKPT initiale sur le périphérique récepteur et identifie la clé initiale dérivée d'un BDK. Ce champ contient généralement les données de dérivation plus une valeur de compteur mise à zéro. Utilisez IK pour AES DUKPT.

Nom de l'en-tête	Objectif	Validation typique	Remarques
KP	KCV de la clé d'emballage	2 caractères hexadécimaux représentent la méthode KCV (00 pour la X9.24 méthode et 01 pour la méthode CMAC). Suivi d'une valeur KCV qui est généralement de 6 caractères hexadécimaux. Par exemple, 010FA329 représente le KCV de 0FA329 calculé à l'aide de la méthode 01 (CMAC).	Cette valeur est utilisée pour instancier l'utilisation de la clé DUKPT initiale sur le périphérique récepteur et identifie la clé initiale dérivée d'un BDK. Ce champ contient généralement les données de dérivation plus une valeur de compteur mise à zéro. Utilisez IK pour AES DUKPT.
PB	Bloc de rembourrage	caractères ASCII imprimables au hasard	Le service remplit automatiquement ce champ lors de l'exportation pour s'assurer que les en-têtes facultatifs sont des multiples de la longueur du bloc de chiffrement

Exporter des clés asymétriques (RSA)

Pour exporter une clé publique sous forme de certificat, utilisez la get-public-key-certificate commande. Cette commande renvoie :

- Le certificat
- Le certificat racine

Les deux certificats sont encodés en base64.

> **Note**
>
> Cette opération n'est pas inopérante : les appels suivants peuvent générer des certificats différents même en utilisant la même clé sous-jacente.

Example

```
$ aws payment-cryptography get-public-key-certificate \
     --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5dza7xqd6soanjtb
```

```
{
"KeyCertificate": "LS0tLS1CRUdJTi...",
"KeyCertificateChain": "LS0tLS1CRUdJT..."
}
```

Rubriques avancées

Cette section couvre les scénarios et les configurations avancés d'échange de clés.

Rubriques

- Apportez votre propre autorité de certification (BYOCA)
- Échange de clés physiques

Apportez votre propre autorité de certification (BYOCA)

Par défaut, lorsqu'un certificat de clé publique est nécessaire pour les clés asymétriques (RSA, ECC) créées au sein du service, ces certificats sont émis par une autorité de chiffrement des AWS paiements et de certification (CA) unique au compte. Cela vise à simplifier son utilisation X.509 sans avoir à identifier ou à configurer une autorité de certification ou à gérer les demandes de signature de certificats (CSR).

AWS La cryptographie des paiements permet également d'utiliser votre propre autorité de certification lorsque cela est nécessaire pour des raisons de politique ou de conformité.

Présentation de

La fonctionnalité BYOCA vous permet d'utiliser votre propre autorité de certification partout où des certificats sont utilisés TR-34 import/export, y compris RSA Unwrap et ECDH-based les transferts de clés. Cela est utile lorsque vous devez maintenir une chaîne de certificats cohérente au sein de votre organisation ou lorsque vous travaillez avec des partenaires qui ont besoin de certificats CA spécifiques. L'exemple suivant illustre le flux de travail BYOCA utilisant l'exportation de TR-34 clés.

Les trois principales différences par rapport au flux TR-34 d'exportation standard sont les suivantes :

1. La clé RSA de signature est explicitement créée à l'aide CreateKeyde. Auparavant, il était créé implicitement via GetParametersForExport.
2. Une nouvelle API GetCertificateSigningRequestcrée une demande de signature de certificat (CSR) qui peut être signée par votre autorité de certification externe.
3. L'ExportKeyAPI est étendue pour permettre la fourniture d'un certificat lors de l'exécution. Auparavant, ce champ était fourni implicitement par`import-token`, ce qui devient un champ facultatif.

⚠ Importantes considérations

- Ces exemples utilisent des RSA-2048 clés et enroulent une TDES-2KEY clé. Lors de l'exportation AES-128, assurez-vous que toutes les clés sont RSA-3072 ou RSA-4096.
- L'erreur la plus courante est que les clés représentées par `SigningKeyIdentifier` et `SigningKeyCertificate` ne correspondent pas.

Flux de travail BYOCA

Les étapes suivantes illustrent le flux de travail BYOCA complet pour TR-34 l'exportation.

Étapes

- Étape 1 : créer une clé RSA
- Étape 2 : générer une demande de signature de certificat
- Étape 3 : Passez en revue le CSR (facultatif)
- Étape 4 : signer le CSR auprès d'une autorité de certification
- Étape 5 : Importer un certificat CA

- Étape 6 : Obtenir le certificat de cryptage KRD
- Étape 7 : Exporter la clé avec BYOCA

Étape 1 : créer une clé RSA

Tout d'abord, créez une paire de clés RSA qui sera finalement le certificat de signature KDH. Vous pouvez ajouter des balises pour identifier l'objectif de la clé.

Example Création d'une clé RSA pour la signature

```
$ aws payment-cryptography create-key --exportable \
    --key-attributes
 KeyAlgorithm=RSA_2048,KeyUsage=TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE,KeyClass=ASYMMETRI
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xgmq6fs6uow736uc",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyAlgorithm": "RSA_2048",
            "KeyModesOfUse": {
                "Sign": true
            }
        },
        "KeyCheckValue": "41E3723C",
        "KeyCheckValueAlgorithm": "SHA_1",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY"
    }
}
```

Prenez note du, `KeyArn` car vous en aurez besoin à l'étape suivante.

Étape 2 : générer une demande de signature de certificat

Générez une demande de signature de certificat (CSR) à signer par votre autorité de certification externe à l'aide de l'GetCertificateSigningRequestAPI. Le résultat est un fichier PEM codé en base64.

Si vous décodez le contenu en base64 et que vous l'enregistrez, vous aurez un CSR valide au format PEM.

Example Générer un CSR

```
$ aws payment-cryptography-data get-certificate-signing-request \
    --key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xgmq6fs6uow736uc \
    --signing-algorithm SHA512 \
    --certificate-subject '{
        "CommonName": "MyCertificateAWSUSEAST",
        "Organization": "Amazon",
        "OrganizationUnit": "PaymentCryptography",
        "Country": "US",
        "StateOrProvince": "Virginia",
        "City": "Arlington"
    }'
```

```
{
    "CertificateSigningRequest": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..."
}
```

Le `CertificateSigningRequest` champ contient le CSR codé en base64 que vous allez envoyer à votre autorité de certification pour signature.

Étape 3 : Passez en revue le CSR (facultatif)

Vous pouvez éventuellement utiliser OpenSSL pour examiner le contenu de la CSR et vous assurer qu'il est valide et conforme aux attentes.

Example Passez en revue la CSR avec OpenSSL

```
$ echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d | openssl req -
text
```

Étape 4 : signer le CSR auprès d'une autorité de certification

Après avoir généré le CSR, vous devez le faire signer par une autorité de certification (CA). Dans les environnements de production, vous utilisez Autorité de certification privée AWS généralement l'infrastructure CA établie de votre entreprise. À des fins de test, vous pouvez utiliser OpenSSL pour créer un certificat auto-signé.

En utilisant Autorité de certification privée AWS

Pour signer le CSR en utilisant Autorité de certification privée AWS, décodez d'abord le CSR codé en base64 et enregistrez-le dans un fichier, puis utilisez l'API. IssueCertificate

Example Signez CSR avec AWS CA privée

```
$ echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d > csr.pem

$ aws acm-pca issue-certificate \
    --certificate-authority-arn arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012 \
    --csr fileb://csr.pem \
    --signing-algorithm SHA256WITHRSA \
    --validity Value=365,Type=DAYS
```

```
{
    "CertificateArn": "arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012/certificate/abcdef1234567890"
}
```

Récupérez ensuite le certificat signé :

Example Récupérez le certificat signé

```
$ aws acm-pca get-certificate \
    --certificate-authority-arn arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012 \
    --certificate-arn arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012/certificate/abcdef1234567890
```

```
{
    "Certificate": "-----BEGIN CERTIFICATE-----\nMIID...\n-----END CERTIFICATE-----",
    "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIID...\n-----END
 CERTIFICATE-----"
}
```

Enregistrez le contenu du certificat pour l'utiliser lors de l'étape d'exportation. Vous devrez l'encoder en base64 lorsque vous le fournirez à l'API. `ExportKey`

Utilisation d'OpenSSL pour les tests

À des fins de test, vous pouvez utiliser OpenSSL pour créer une autorité de certification autosignée et signer le CSR. Créez d'abord une clé privée CA et un certificat auto-signé :

Example Création d'une autorité de certification de test avec OpenSSL

```
$ # Generate CA private key
openssl genrsa -out ca-key.pem 4096

$ # Create self-signed CA certificate
openssl req -new -x509 -days 3650 -key ca-key.pem -out ca-cert.pem \
    -subj "/C=US/ST=Virginia/L=Arlington/O=TestOrg/CN=Test CA"
```

Décodez ensuite le CSR de l'étape précédente et signez-le avec votre autorité de certification de test :

Example Signer un CSR avec OpenSSL

```
$ # Decode the base64-encoded CSR
echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d > csr.pem

$ # Sign the CSR with the CA
openssl x509 -req -in csr.pem -CA ca-cert.pem -CAkey ca-key.pem \
    -CAcreateserial -out signed-cert.pem -days 365 -sha512
```

```
Certificate request self-signature ok
subject=C=US, ST=Virginia, L=Arlington, O=Amazon, OU=PaymentCryptography,
 CN=MyCertificateAWSUSEAST
```

Le certificat signé est maintenant disponible`signed-cert.pem`. Vous devez encoder ce certificat en base64 lorsque vous le fournissez à l'API : `ExportKey`

Example Coder le certificat signé en Base64

```
$ cat signed-cert.pem | base64 -w 0
```

Étape 5 : Importer un certificat CA

Toute autorité de certification utilisée doit d'abord être fiable pour empêcher l'utilisation de certificats arbitraires. Importez le certificat racine de votre autorité de certification externe à l'aide de

l'ImportKeyAPI. Si vous utilisez une autorité de certification intermédiaire, appelez `import-key` à nouveau mais spécifiez à la `TrustedPublicKey` place `RootCertificatePublicKey` et spécifiez l'ARN de l'autorité de certification racine.

Example Importer le certificat Root CA

```
$ aws payment-cryptography import-key --key-material='{
    "RootCertificatePublicKey": {
        "KeyAttributes": {
            "KeyAlgorithm": "RSA_4096",
            "KeyClass": "PUBLIC_KEY",
            "KeyModesOfUse": {
                "Verify": true
            },
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
        },
        "PublicKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t..."
    }
}'
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xivpaqy7qbbm7cdw",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096",
            "KeyModesOfUse": {
                "Verify": true
            }
        },
        "Enabled": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL"
    }
}
```

Prenez note des CA à utiliser `KeyArn` lors de l'étape d'exportation.

Étape 6 : Obtenir le certificat de cryptage KRD

Dans cet exemple, nous réimportons dans AWS Payment Cryptography. Nous appelons donc le service pour recevoir un certificat de clé publique KRD à l'aide de l'GetParametersForImportAPI. Dans un scénario réel, cela serait fourni par l'autre système, comme un HSM, un guichet automatique, un terminal de paiement ou un système de gestion de terminaux de paiement.

Example Obtenir les paramètres pour l'importation

```
$ aws payment-cryptography-data get-parameters-for-import \
    --key-material-type "TR34_KEY_BLOCK" \
    --wrapping-key-algorithm RSA_2048
```

```
{
    "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "WrappingKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "WrappingKeyAlgorithm": "RSA_2048",
    "ImportToken": "import-token-v2rxpl6drxeptn7w",
    "ParametersValidUntilTimestamp": "2025-11-01T18:45:31.271000-07:00"
}
```

Étape 7 : Exporter la clé avec BYOCA

Enfin, exportez la clé à l' TR-34 aide de votre propre CA-signed certificat à l'aide de l'ExportKeyAPI. Fournissez le certificat de signature signé par votre autorité de certification externe.

Example TR-34 Exporter avec BYOCA

```
$ aws payment-cryptography-data export-key \
    --export-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
iox73p5f4c4yjiod \
    --key-material '{
        "Tr34KeyBlock": {
            "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-1:111122223333:key/j625deyfqlwctu57",
            "SigningKeyIdentifier": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/xgmq6fs6uow736uc",
            "SigningKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
            "KeyBlockFormat": "X9_TR34_2012",
            "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t..."
        }
    }'
```

```
{
    "WrappedKey": {
        "WrappedKeyMaterialFormat": "TR34_KEY_BLOCK",
        "KeyMaterial": "3082055A06092A864886F70D010702A082054B30820547...",
        "KeyCheckValue": "3DCA31",
        "KeyCheckValueAlgorithm": "ANSI_X9_24"
    }
}
```

Le bloc-clé exporté peut désormais être importé par le système récepteur à l'aide du processus TR-34 d'importation standard.

Remarques supplémentaires

- Ces exemples sont présentés à l'aide de l'interface de ligne de commande AWS. Les mêmes fonctionnalités sont disponibles dans tous les kits SDK AWS, notamment Java, Python, Go et Rust.
- Si vous testez avec une autorité de certification autosignée, vous pouvez utiliser OpenSSL pour créer une autorité de certification de test et signer le CSR. En production, utilisez l'infrastructure CA établie de votre organisation.

Échange de clés physiques

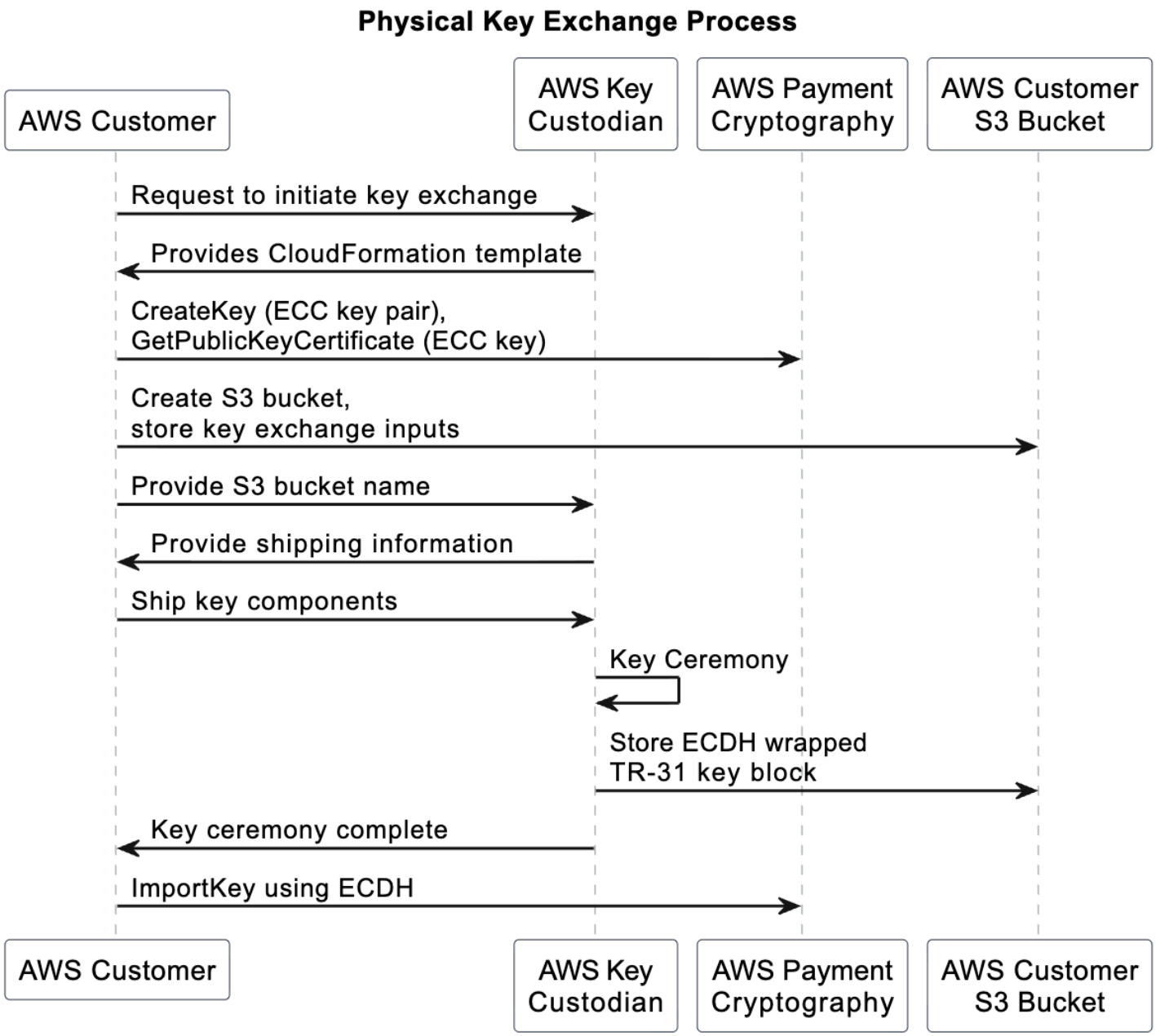
Vous pouvez utiliser l'échange de clés physiques pour convertir en toute sécurité des composants clés cryptographiques sur support papier au format électronique lorsque vos partenaires ou fournisseurs ne prennent pas en charge l'échange électronique de clés. Des dépositaires de AWS clés qualifiés organisent des cérémonies clés dans des installations AWS-operated sécurisées certifiées PCI PIN et P2PE, convertissant les composants clés papier en format électronique à l'aide d'un HSM hors ligne. Le service utilise l'échange de ECDH-based clés pour délivrer un bloc ECDH-wrapped TR-31 clé, que vous importez directement dans votre compte AWS Payment Cryptography.

> **Note**
>
> Nous recommandons d'utiliser des normes dans la mesure du Importation et exportation de clés possible. Utilisez l'échange de clés physiques uniquement lorsque vos partenaires ou fournisseurs ne prennent pas en charge les méthodes d'échange de clés électroniques telles que l'ANSI X9.24 TR-34, le RSA ou l' wrap/unwrapECDH.

Comment fonctionne l'échange physique de clés

Pour lancer l'échange de clés papier, un CloudFormation modèle vous guide tout au long de la configuration préalable, y compris la création d'une paire de clés ECC et d'un compartiment S3 dans votre compte. Vous ou votre partenaire expédiez ensuite les composants clés en papier à l'installation AWS sécurisée, où des dépositaires de AWS clés qualifiés effectuent la cérémonie de remise des clés à l'aide d'un HSM hors ligne. Le résultat est un bloc ECDH-wrapped TR-31 clé chargé dans votre compartiment S3, que vous importez dans votre compte à l'aide de Importation de clés à l'aide de techniques asymétriques (ECDH) cette méthode. Physical Key Exchange prend en charge l'importation de clés KEK (utilisation des clés K1) ou BDK (utilisation des clés B0) dans les algorithmes de clés TDES et AES.

Le schéma suivant montre le processus d'échange de clés physiques de bout en bout.



1. Initiation — Vous soumettez un ticket d'assistance ou vous contactez votre responsable de compte pour soumettre une demande.
2. Configuration du client — AWS Payment Cryptography fournit un CloudFormation modèle vous permettant de suivre les étapes préalables suivantes :
   - Créez une paire de clés ECC P521 dans votre compte AWS Payment Cryptography et récupérez le certificat de clé publique.

- Créez un compartiment Amazon S3 avec une politique accordant l' read/write accès principal au service AWS Payment Cryptography.
- Stockez le certificat public ECC et l'autorité de certification racine de signature dans le compartiment Amazon S3.
- Indiquez les principaux attributs : utilisation des clés, principaux modes d'utilisation et nombre de composants clés en papier à envoyer.

3. Partager le nom du compartiment S3 : le client partage le nom du compartiment S3 créé par la CloudFormation pile, où le certificat de clé publique, la chaîne de certificats et les attributs clés sont stockés pour que AWS Payment Cryptography lance l'échange de clés.
4. Coordination de l'expédition — La cryptographie des AWS paiements fournit les détails d'expédition pour l'installation US-based sécurisée. Vous ou votre partenaire expédiez les composants clés en papier aux AWS principaux dépositaires.
5. Réception des composants : les AWS principaux dépositaires reçoivent chaque composant papier et envoient un accusé de réception distinct pour chaque composant.
6. Cérémonie des AWS clés : les gardiens des clés exécutent la cérémonie des clés à l'aide d'un HSM hors ligne. Le bloc de TR-31 clé obtenu, encapsulé à l'aide d'une ECDH-derived AES-256 clé, le certificat public ECC du HSM hors ligne et son certificat de signature sont chargés dans votre compartiment Amazon S3.
7. Achèvement — La cryptographie des AWS paiements envoie une confirmation indiquant que la cérémonie des clés est terminée. Vous pouvez ensuite importer le bloc de TR-31 clé encapsulé ECDH dans votre compte AWS Payment Cryptography à l'Importation de clés à l'aide de techniques asymétriques (ECDH)aide de cette méthode.
8. Facturation — Vous êtes facturé par clé échangée une fois la cérémonie des clés terminée avec succès.

Conformité et sécurité

L'échange de clés physiques fonctionne dans des installations AWS sécurisées conçues pour répondre aux exigences de sécurité physique et logique PCI PIN et PCI P2PE. Les contrôles suivants sont en place :

Double contrôle et séparation des tâches

AWS les principaux dépositaires sont désignés par différentes équipes dotées de structures hiérarchiques distinctes. Des processus sont en place pour garantir que les principales étapes des cérémonies sont effectuées sous double contrôle.

HSM hors ligne

Les cérémonies clés sont effectuées à l'aide de modules de sécurité HSM-listed matériels certifiés PCI PTS qui fonctionnent hors ligne sans connexion réseau. Votre clé n'existe jamais en texte clair en dehors des limites du HSM.

Livraison de clés cryptographiques

Les informations clés sont transférées du HSM hors ligne vers votre compte AWS Payment Cryptography par échange de ECDH-based clés, garantissant ainsi une protection cryptographique de bout en bout.

Audit et conformité

AWS a mis en place des processus pour répondre aux exigences de conformité applicables qui sont évaluées périodiquement pour les attestations PCI PIN et P2PE. Consultez le package de conformité dans AWS Artifact pour les rapports auxquels vous pouvez faire référence dans vos propres évaluations PCI.

Utilisation des alias

Un alias est un nom convivial pour une clé AWS de cryptographie de paiement. Par exemple, un alias vous permet de faire référence à une clé comme `alias/test-key` au lieu de`arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h`.

Vous pouvez utiliser un alias pour identifier une clé dans la plupart des opérations de gestion des clés (plan de contrôle) et dans les opérations cryptographiques (plan de données).

Vous pouvez également autoriser et refuser l'accès à la clé AWS de chiffrement des paiements en fonction de leurs alias sans modifier les politiques ni gérer les subventions. Cette fonctionnalité fait partie du support du service pour le contrôle d'accès basé sur les attributs (ABAC).

La puissance des alias provient en grande partie de votre capacité à modifier la clé associée à un alias à tout moment. Les alias peuvent faciliter l'écriture et la maintenance de votre code. Supposons, par exemple, que vous utilisiez un alias pour faire référence à une clé de cryptographie de AWS paiement particulière et que vous souhaitiez modifier la clé de cryptographie AWS de paiement. Dans ce cas, associez simplement l'alias à une autre clé. Il n'est pas nécessaire de modifier le code ou la configuration de l'application.

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 de

cryptographie de AWS paiement dans sa région. Lorsque le code s'exécute dans chaque région, l'alias fait référence à la clé de cryptographie de AWS paiement associée dans cette région.

Vous pouvez créer un alias pour une clé AWS de chiffrement des paiements à l'aide de l'`CreateAlias`API.

L'API AWS de cryptographie des paiements 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 le KeyARN lié (list-aliases), de modifier la clé de chiffrement des AWS paiements associée à un alias (update-alias) et de supprimer un alias (delete-alias).

Rubriques

À propos des alias

Découvrez comment fonctionnent les alias dans la cryptographie des AWS paiements.

Un alias est une AWS ressource indépendante

Un alias n'est pas une propriété d'une clé de chiffrement des AWS paiements. Les actions que vous effectuez sur l'alias n'affectent pas la clé qui lui est associée. Vous pouvez créer un alias pour une clé de cryptographie de AWS paiement, puis mettre à jour l'alias afin qu'il soit associé à une autre clé de cryptographie AWS de paiement. Vous pouvez même supprimer l'alias sans aucun effet sur la clé de cryptographie AWS de paiement associée. Si vous supprimez une clé AWS de chiffrement des paiements, tous les alias associés à cette clé ne seront plus attribués.

Si vous spécifiez un alias comme ressource dans une politique IAM, celle-ci fait référence à l'alias, et non à la clé de chiffrement des AWS paiements associée.

Chaque alias possède un nom convivial

Lorsque vous créez un alias, vous spécifiez le nom de l'alias préfixé par`alias/`. Par exemple `alias/test_1234`

Chaque alias est associé à une clé AWS de cryptographie de paiement à la fois

L'alias et sa clé AWS de cryptographie de paiement doivent se trouver dans le même compte et dans la même région.

Une clé AWS de cryptographie de paiement peut être associée à plusieurs alias simultanément, mais chaque alias ne peut être associé qu'à une seule clé

Par exemple, cette `list-aliases` sortie indique que l'`alias/sampleAlias1`alias est associé à une seule clé de cryptographie de AWS paiement cible, qui est représentée par la `KeyArn` propriété.

```
$ aws payment-cryptography list-aliases
```

```
  {
    "Aliases": [
        {
            "AliasName": "alias/sampleAlias1",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
        }
    ]
}
```

Plusieurs alias peuvent être associés à la même clé de cryptographie AWS de paiement

Par exemple, vous pouvez associer les `alias/sampleAlias2` alias `alias/sampleAlias1;` et à la même clé.

```
$ aws payment-cryptography list-aliases
```

```
{
        "Aliases": [
            {
                "AliasName": "alias/sampleAlias1",
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
            },
            {
                "AliasName": "alias/sampleAlias2",
```

```
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
            }
        ]
    }
```

Un alias doit être unique pour un compte et une région donnés

Par exemple, vous ne pouvez avoir qu'un seul alias `alias/sampleAlias1` dans chaque compte et région. Les alias font la distinction majuscules/majuscules, mais nous vous déconseillons d'utiliser des alias dont les majuscules ne diffèrent que par leur mise en majuscules, car ils peuvent être sujets à des 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 `alias/sampleAlias2` dans l'est des États-Unis (Virginie du Nord) et un alias `alias/sampleAlias2` dans l'ouest des États-Unis (Oregon). Chaque alias serait associé à une clé de cryptographie de AWS paiement 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 en utilise un différent alias/sampleAlias2. Pour en savoir plus, consultez Utilisation d'alias dans vos applications.

Vous pouvez modifier la clé AWS de chiffrement des paiements associée à un alias

Vous pouvez utiliser cette `UpdateAlias` opération pour associer un alias à une autre clé de cryptographie de AWS paiement. Par exemple, si l'`alias/sampleAlias2`alias est associé à la clé de chiffrement des `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` AWS paiements, vous pouvez le mettre à jour afin qu'il soit associé à la `arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi` clé.

> **Warning**
>
> AWS La cryptographie des paiements ne permet pas de vérifier que les anciennes et les nouvelles clés possèdent toutes les mêmes attributs, tels que l'utilisation des clés. La mise à jour avec un autre type de clé peut entraîner des problèmes dans votre application.

Certaines clés n'ont pas d'alias

Un alias est une fonctionnalité facultative et toutes les clés n'auront pas d'alias, sauf si vous choisissez de gérer votre environnement de cette manière. Les clés peuvent être associées à des alias à l'aide de la `create-alias` commande. Vous pouvez également utiliser l'opération update-alias pour modifier la clé de chiffrement des AWS paiements associée à un alias et l'opération delete-alias pour supprimer un alias. Par conséquent, certaines clés de chiffrement des AWS paiements peuvent avoir plusieurs alias, tandis que d'autres n'en ont aucun.

Associer une clé à un alias

Vous pouvez mapper une clé (représentée par un ARN) à un ou plusieurs alias à l'aide de la `create-alias` commande. Cette commande n'est pas idempotente. Pour mettre à jour un alias, utilisez la commande update-alias.

```
$ aws payment-cryptography create-alias --alias-name alias/sampleAlias1 \
             --key-arn arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h
```

```
{
    "Alias": {
        "AliasName": "alias/alias/sampleAlias1",
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
    }
}
```

Utilisation d'alias dans vos applications

Vous pouvez utiliser un alias pour représenter une clé de chiffrement des AWS paiements dans le code de votre application. Le `key-identifier` paramètre utilisé dans les opérations relatives aux données de cryptographie des AWS paiements ainsi que dans d'autres opérations telles que les clés de liste accepte un alias ou un alias ARN.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier alias/
BIN_123456_CVK --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue2={CardExpiryDate=0123}
```

Lorsque vous utilisez un alias ARN, n'oubliez pas que le mappage d'alias vers une clé de cryptographie de AWS paiement est défini dans le compte propriétaire de la clé de cryptographie de AWS paiement et peut varier d'une région à l'autre.

L'une des utilisations les plus performantes des alias est au niveau des applications qui s'exécutent dans plusieurs Régions AWS.

Vous pouvez créer une version différente de votre application dans chaque région ou utiliser un dictionnaire, une configuration ou une instruction switch pour sélectionner la clé de cryptographie de AWS paiement adaptée à chaque région. Mais il peut être plus facile de créer un alias portant le même nom d'alias dans chaque région. Rappelez-vous que ne nom de l'alias est sensible à la casse.

API associées

Balises

Les balises sont des paires clé/valeur qui agissent comme des métadonnées pour organiser vos clés AWS de cryptographie de paiement. Ils peuvent être utilisés pour identifier les clés de manière flexible ou pour regrouper une ou plusieurs clés.

Obtenez des clés

Une clé AWS de cryptographie de paiement représente une unité unique de matériel cryptographique et ne peut être utilisée que pour les opérations cryptographiques de ce service. L' GetKeys API prend une entrée et renvoie KeyIdentifier les métadonnées clés, y compris les attributs, l'état et les horodatages, mais ne renvoie pas le contenu de la clé cryptographique réelle.

Example

```
$ aws payment-cryptography get-key --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```


```
{
  "Key": {
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
      "KeyAttributes": {
          "KeyUsage": "TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY",
          "KeyClass": "SYMMETRIC_KEY",
          "KeyAlgorithm": "AES_128",
          "KeyModesOfUse": {
              "Encrypt": true,
              "Decrypt": true,
              "Wrap": true,
              "Unwrap": true,
              "Generate": false,
              "Sign": false,
              "Verify": false,
              "DeriveKey": false,
              "NoRestrictions": false
          }
      },
      "KeyCheckValue": "0A3674",
      "KeyCheckValueAlgorithm": "CMAC",
      "Enabled": true,
      "Exportable": true,
      "KeyState": "CREATE_COMPLETE",
      "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
      "CreateTimestamp": "2023-06-02T07:38:14.913000-07:00",
      "UsageStartTimestamp": "2023-06-02T07:38:14.857000-07:00"
  }
}
```

key/certificate Associer le public à une paire de clés

Get Public Key/Certificate renvoie la clé publique indiquée par le`KeyArn`. Il peut s'agir de la partie clé publique d'une paire de clés générée par AWS Payment Cryptography ou d'une clé publique précédemment importée. Le cas d'utilisation le plus courant consiste à fournir la clé publique à un service externe qui cryptera les données. Ces données peuvent ensuite être transmises à une application utilisant la cryptographie des AWS paiements et les données peuvent être déchiffrées à l'aide de la clé privée sécurisée dans la cryptographie des paiements. AWS

Le service renvoie les clés publiques sous forme de certificat public. Le résultat de l'API contient l'autorité de certification et le certificat de clé publique. Les deux éléments de données sont codés en base64.

> **Note**
>
> Le certificat public renvoyé est destiné à être de courte durée et n'est pas destiné à être idempotent. Vous pouvez recevoir un certificat différent à chaque appel d'API, même si la clé publique elle-même reste inchangée.

Example

```
$ aws payment-cryptography get-public-key-certificate  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/nsq2i3mbg6sn775f
```

```
{
  "KeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2VENDQXFXZ0F3SUJBZ0lSQUo1OWd2VkpDd3dlYldMNldYZEpYY
  "KeyCertificateChain":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUY0VENDQThtZ0F3SUJBZ0lSQUtlN2piaHFKZjJPd3FGUWI5c3VuC
}
```

Clés de balisage

Dans la cryptographie des AWS paiements, vous pouvez ajouter des balises à une clé de chiffrement des AWS paiements lorsque vous créez une clé, et étiqueter ou décocher les clés existantes, sauf si elles sont en attente de suppression. Les balises sont facultatives, mais peuvent être très utiles.

Pour obtenir des informations générales sur les balises, notamment les meilleures pratiques, les stratégies de balisage, ainsi que le format et la syntaxe des balises, consultez les AWS ressources de balisage dans le. Référence générale d'Amazon Web Services

Rubriques

À propos des balises dans la cryptographie des AWS paiements

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 Payment Cryptography, vous pouvez ajouter des balises à une clé lorsque vous créez la clé, et étiqueter ou décocher les clés existantes, sauf si elles sont en attente de suppression. Vous ne pouvez pas baliser les alias. Les balises sont facultatives, mais peuvent être très utiles.

Par exemple, vous pouvez ajouter une `"Project"="Alpha"` balise à toutes les clés de chiffrement des AWS paiements et à tous les compartiments Amazon S3 que vous utilisez pour le projet Alpha. Un autre exemple consiste à ajouter une `"BIN"="20130622"` étiquette à toutes les clés associées à un numéro d'identification bancaire (BIN) spécifique.

```
    [
    {
      "Key": "Project",
      "Value": "Alpha"
    },
    {
      "Key": "BIN",
      "Value": "20130622"
    }
  ]
```

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 de cryptographie de AWS paiement 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 d'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 les coûts AWS de cryptographie des paiements pour un projet, une application ou un centre de coûts.

  Pour en savoir plus sur l'utilisation des balises pour l'allocation des coûts, veuillez consulter Utilisation des balises de répartition des coûts dans le Guide de l'utilisateur AWS Billing . Pour plus d'informations sur les règles relatives aux clés de balise et aux valeurs des balises, consultez la section Restrictions relatives aux User-Defined balises dans le guide de AWS Billing l'utilisateur.
- Contrôlez l'accès à vos AWS ressources. L'autorisation et le refus d'accès aux clés en fonction de leurs balises font partie de la prise en charge de la cryptographie des AWS paiements pour le contrôle d'accès basé sur les attributs (ABAC). Pour plus d'informations sur le contrôle de l'accès à la cryptographie des AWS paiements en fonction de leurs balises, consultezAutorisation basée sur les balises AWS de cryptographie des paiements. 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 La cryptographie des paiements écrit une entrée dans votre AWS CloudTrail journal lorsque vous utilisez les ListTagsForResource opérations TagResource UntagResource, ou.

Afficher les tags clés dans la console

Pour afficher les balises dans la console, vous devez disposer d'une autorisation de balisage sur la clé issue d'une politique IAM qui inclut la clé. Vous avez besoin de ces autorisations en plus des autorisations pour afficher les clés dans la console.

Gestion des balises clés à l'aide des opérations d'API

Vous pouvez utiliser l'API AWS Payment Cryptography pour ajouter, supprimer et répertorier les balises des clés que vous gérez. Ces exemples utilisent l'AWS Command Line Interface (AWS CLI), mais vous pouvez utiliser n'importe quel langage de programmation pris en charge. Vous ne pouvez pas étiqueter Clés gérées par AWS.

Pour ajouter, modifier, afficher et supprimer des balises pour une clé, vous devez disposer des autorisations requises. Pour en savoir plus, consultez Contrôle de l'accès aux balises.

Rubriques

- CreateKey: Ajouter des balises à une nouvelle clé
- TagResource: ajouter ou modifier les balises d'une clé
- ListResourceTags: Récupère les tags d'une clé
- UntagResource: Supprimer les tags d'une clé

CreateKey: Ajouter des balises à une nouvelle clé

Vous pouvez ajouter des balises lorsque vous créez une clé. Pour définir les balises, utilisez le `Tags` paramètre de l'CreateKeyopération.

Pour ajouter des balises lors de la création d'une clé, l'appelant doit être `payment-cryptography:TagResource` autorisé dans une politique IAM. L'autorisation doit au minimum couvrir toutes les clés du compte et de la région. Pour en savoir plus, consultez Contrôle de l'accès aux balises.

La valeur du paramètre `Tags` de `CreateKey` est une collection de paires de clés et de valeurs de balises sensibles à la casse. Chaque étiquette d'une clé doit porter 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 de chiffrement symétrique avec une `Project:Alpha` balise. Lorsque vous spécifiez plusieurs paires clé-valeur, utilisez un espace pour séparer chaque paire.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY, \
       KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY, \
       KeyModesOfUse='{Generate=true,Verify=true}' \
       --tags '[{"Key":"Project","Value":"Alpha"},{"Key":"BIN","Value":"123456"}]'
```

Lorsque cette commande aboutit, elle renvoie un `Key` objet contenant des informations sur la nouvelle clé. Cependant, l'objet `Key` n'inclut pas les balises. Pour obtenir les balises, utilisez l'ListResourceTagsopération.

TagResource: ajouter ou modifier les balises d'une clé

L'TagResourceopération ajoute une ou plusieurs balises à une clé. Vous ne pouvez pas utiliser cette opération pour ajouter ou modifier des balises dans un autre Compte AWS.

Pour ajouter une balise, spécifiez de nouvelles clé et valeur de balises. Pour modifier une balise, spécifiez une clé de balise existante et une nouvelle valeur de balise. Chaque étiquette d'une clé doit avoir une clé de balise différente. La valeur de balise peut être une chaîne vide ou nulle.

Par exemple, la commande suivante ajoute **`UseCase`** des **`BIN`** balises à un exemple de clé.

```
$ aws payment-cryptography tag-resource --resource-arn arn:aws:payment-
cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h --tags
 '[{"Key":"UseCase","Value":"Acquiring"},{"Key":"BIN","Value":"123456"}]'
```

Lorsque cette commande aboutit, elle ne renvoie pas de sortie. Pour afficher les balises d'une clé, utilisez l'ListResourceTagsopération.

Vous pouvez également utiliser TagResource pour modifier la valeur de balise d'une balise existante. Pour remplacer une valeur de balise, spécifiez la même clé de balise avec une valeur différente. Les balises non répertoriées dans une commande de modification ne sont ni modifiées ni supprimées.

Par exemple, cette commande modifie la valeur de la balise `Project` de `Alpha` en `Noe`.

La commande sera http/200 renvoyée sans contenu. Pour voir vos modifications, utilisez `ListTagsForResource`

```
$ aws payment-cryptography tag-resource --resource-arn arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h \
        --tags '[{"Key":"Project","Value":"Noe"}]'
```

ListResourceTags: Récupère les tags d'une clé

L'ListResourceTagsopération obtient les balises d'une clé. Le paramètre `ResourceArn` (KeyArn ou KeyAlias) est obligatoire. Vous ne pouvez pas utiliser cette opération pour afficher les balises des touches d'une autre manière Compte AWS.

Par exemple, la commande suivante obtient les balises d'un exemple de clé.

```
$ aws payment-cryptography list-tags-for-resource --resource-arn arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h

  {
    "Tags": [
        {
            "Key": "BIN",
            "Value": "20151120"
        },
        {
            "Key": "Project",
            "Value": "Production"
        }
    ]
}
```

UntagResource: Supprimer les tags d'une clé

L'UntagResourceopération supprime les balises d'une clé. Pour identifier les balises à supprimer, spécifiez les clés de balise. Vous ne pouvez pas utiliser cette opération pour supprimer des balises d'une autre clé Compte AWS.

Lorsque l'opération `UntagResource` réussit, elle ne renvoie aucune sortie. De plus, si la clé de balise spécifiée n'est pas trouvée sur la clé, elle ne lance pas d'exception ni ne renvoie de réponse. Pour vérifier que l'opération a fonctionné, ListResourceTagsutilisez-la.

Par exemple, cette commande supprime le **`Purpose`** tag et sa valeur de la clé spécifiée.

```
$ aws payment-cryptography untag-resource \
        --resource-arn arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h --tag-keys Project
```

Contrôle de l'accès aux balises

Pour ajouter, afficher et supprimer des balises à l'aide de l'API, les directeurs doivent disposer d'autorisations de balisage dans les politiques IAM.

Vous pouvez également limiter ces autorisations en utilisant des clés de condition AWS globales pour les balises. Dans le AWS domaine de la cryptographie des paiements, ces conditions peuvent contrôler l'accès aux opérations de marquage, telles que TagResourceet. UntagResource

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.

cryptographie des paiements : TagResource

Autorise les principaux à ajouter ou à modifier des balises. Pour ajouter des balises lors de la création d'une clé, le principal doit disposer d'une autorisation dans le cadre d'une politique IAM qui n'est pas limitée à des clés spécifiques.

cryptographie des paiements : ListTagsForResource

Permet aux principaux d'afficher les balises sur les clés.

cryptographie des paiements : UntagResource

Permet aux principaux de supprimer les balises des clés.

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 de taguer la clé. Il accorde à tous les utilisateurs qui peuvent endosser les rôles Administrateur ou Développeur d'exemple l'autorisation d'afficher les balises.

JSON

```
{
  "Version":"2012-10-17",
  "Id": "example-key-policy",
  "Statement": [
    {
      "Sid": "EnableIAMUserPermissions",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
      "Action": "payment-cryptography:*",
      "Resource": "*"
    },
    {
      "Sid": "AllowAllTaggingPermissions",
      "Effect": "Allow",
      "Principal": {"AWS": [
        "arn:aws:iam::111122223333:user/LeadAdmin",
        "arn:aws:iam::111122223333:user/SupportLead"
      ]},
      "Action": [
        "payment-cryptography:TagResource",
        "payment-cryptography:ListTagsForResource",
        "payment-cryptography: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": "payment-cryptography:ListTagsForResource",
      "Resource": "*"
    }
  ]
}
```

Pour autoriser les principaux à baliser plusieurs clés, vous pouvez utiliser une politique IAM. Pour que cette politique soit efficace, la politique clé de chaque clé doit autoriser le compte à utiliser les politiques IAM pour contrôler l'accès à la clé.

Par exemple, la politique IAM suivante permet aux principaux de créer des clés. Cela leur permet également de créer et de gérer des balises sur toutes les clés du compte spécifié. Cette combinaison permet aux principaux d'utiliser le paramètre tags de l'CreateKeyopération pour ajouter des balises à une clé lors de sa création.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKeys",
      "Effect": "Allow",
      "Action": "payment-cryptography:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyTags",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:TagResource",
        "payment-cryptography:UntagResource",
        "payment-cryptography:ListTagsForResource"
      ],
      "Resource": "arn:aws:payment-cryptography:*: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 `payment-cryptography:TagResource` et `payment-cryptography: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.

- lois : RequestTag
- aws :ResourceTag/tag-key (politiques IAM uniquement)
- lois : TagKeys

Lorsque vous utilisez des balises pour contrôler l'accès aux clés, il est recommandé d'utiliser la touche de `aws:TagKeys` condition `aws:RequestTag/`*`tag-key`* ou 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.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKey",
      "Effect": "Allow",
      "Action": "payment-cryptography:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyViewAllTags",
      "Effect": "Allow",
```

```
      "Action": "payment-cryptography:ListTagsForResource",
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*"
    },
    {
      "Sid": "IAMPolicyManageTags",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:TagResource",
        "payment-cryptography:UntagResource"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
          "ForAllValues:StringEquals": {"aws:TagKeys": "Project"}
      }
    }
  ]
}
```

Utilisation de balises pour contrôler l'accès aux clés

Vous pouvez contrôler l'accès à la cryptographie des AWS paiements en fonction des balises figurant sur la clé. Par exemple, vous pouvez écrire une politique IAM qui permet aux principaux d'activer et de désactiver uniquement les clés dotées d'une balise particulière. Vous pouvez également utiliser une politique IAM pour empêcher les principaux d'utiliser des clés dans des opérations cryptographiques, à moins que la clé ne comporte une balise particulière.

Cette fonctionnalité fait partie de la prise en charge de la cryptographie des AWS paiements pour le 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.

AWS La cryptographie des paiements prend en charge la clé contextuelle de condition globale aws :ResourceTag/tag-key, qui vous permet de contrôler l'accès aux clés en fonction des balises figurant sur la clé. Comme plusieurs clés peuvent avoir le même tag, cette fonctionnalité vous permet d'appliquer l'autorisation à un ensemble de clés sélectionné. Vous pouvez également modifier facilement les clés de l'ensemble en modifiant leurs balises.

Dans la cryptographie des AWS paiements, 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, ni dans les opérations qui n'utilisent pas de clé 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 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 aux responsables IAM uniquement les autorisations dont ils ont besoin sur les clés qu'ils doivent utiliser ou gérer. Par exemple, utilisez des balises pour étiqueter les clés utilisées pour un projet. Donnez ensuite à l'équipe du projet l'autorisation d'utiliser uniquement les clés portant le tag du projet.
- Soyez prudent lorsque vous donnez aux principaux les autorisations `payment-cryptography:TagResource` et `payment-cryptography:UntagResource` qui leur permettent d'ajouter, de modifier et de supprimer des balises. Lorsque vous utilisez des balises pour contrôler l'accès aux clés, la modification d'une balise peut autoriser les principaux à utiliser des clés qu'ils n'étaient pas autorisés à utiliser autrement. Il peut également refuser l'accès aux clés dont les autres directeurs ont besoin pour faire leur travail. Les administrateurs clés qui ne sont pas autorisés à modifier les politiques clés ou à créer des autorisations peuvent contrôler l'accès aux clés 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 de balisage d'un principal à des balises ou à des modèles de balises spécifiques sur des clés spécifiques.
- Passez en revue les principes Compte AWS qui disposent actuellement d'autorisations de balisage et de débalisage et ajustez-les si nécessaire. Les politiques IAM peuvent autoriser des autorisations de balisage et de débalisage sur toutes les clés. Par exemple, la politique gérée par l'administrateur permet aux principaux de baliser, de débaliser et de répertorier les balises sur toutes les clés.
- Avant de définir une politique qui dépend d'une balise, passez en revue les balises figurant sur les clés 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.
- 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 : Cela permet aux principaux d'appeler les opérations de déchiffrement uniquement sur les clés de votre compte correspondant à la région des États-Unis Est (Virginie du Nord) et dotées d'une `"Project"="Alpha"` étiquette. Vous pouvez attacher cette politique à des rôles dans l'exemple de projet Alpha.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:DecryptData"
      ],
      "Resource": "arn:aws:payment-cryptography:us-east-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

L'exemple de politique IAM suivant permet aux principaux d'utiliser n'importe quelle clé du compte pour certaines opérations cryptographiques. Mais il interdit aux donneurs d'ordre d'utiliser ces opérations cryptographiques sur des clés comportant ou non `"Type"` une `"Type"="Reserved"` étiquette.

JSON

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

```
  "Statement": [
    {
      "Sid": "IAMAllowCryptographicOperations",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:EncryptData",
        "payment-cryptography:DecryptData",
        "payment-cryptography:ReEncrypt*"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*"
    },
    {
      "Sid": "IAMDenyOnTag",
      "Effect": "Deny",
      "Action": [
        "payment-cryptography:EncryptData",
        "payment-cryptography:DecryptData",
        "payment-cryptography:ReEncrypt*"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Type": "Reserved"
        }
      }
    },
    {
      "Sid": "IAMDenyNoTag",
      "Effect": "Deny",
      "Action": [
       "payment-cryptography:EncryptData",
       "payment-cryptography:DecryptData",
       "payment-cryptography:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "Null": {
          "aws:ResourceTag/Type": "true"
        }
      }
    }
  ]
}
```

Comprendre les principaux attributs de la clé AWS de cryptographie des paiements

L'un des principes d'une bonne gestion des clés est que les clés ont une portée appropriée et ne peuvent être utilisées que pour des opérations autorisées. Ainsi, certaines touches ne peuvent être créées qu'avec certains modes d'utilisation des touches. Dans la mesure du possible, cela correspond aux modes d'utilisation disponibles tels que définis par TR-31.

Bien que la cryptographie des AWS paiements vous empêche de créer des clés non valides, des combinaisons valides sont fournies ici pour votre commodité.

Clés symétriques

- TR31_B0_BASE_DERIVATION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}
- TR31_C0_CARD_VERIFICATION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée de modes d'utilisation clés : {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY*, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}
- TR31_E1_EMV_MKEY_CONFIDENTIALITY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}

- TR31_E2_EMV_MKEY_INTEGRITY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}
- TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}
- TR31_E5_EMV_MKEY_CARD_PERSONALIZATION
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}
- TR31_E6_EMV_MKEY_AUTRE
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}, { NoRestrictions = true}
- TR31_K0_KEY_ENCRYPTION_KEY
  - Il est recommandé d'utiliser TR31_K1_KEY_BLOCK_PROTECTION_KEY. Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée de modes d'utilisation clés : {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_K1_KEY_BLOCK_PROTECTION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée de modes d'utilisation clés : {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_M1_ISO_9797_1_MAC_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M3_ISO_9797_3_MAC_KEY

  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M6_ISO_9797_5_CMAC_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M7_HMAC_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_P0_PIN_ENCRYPTION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée de modes d'utilisation clés : {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_V1_IBM3624_PIN_VERIFICATION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_V2_VISA_PIN_VERIFICATION_KEY
  - Algorithmes clés autorisés : TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinaison autorisée des principaux modes d'utilisation : {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}

Clés asymétriques

- TR31_D1_CLÉ_ASYMÉTRIQUE_POUR_CHIFFRE_DONNÉES
  - Algorithmes clés autorisés : RSA_2048, RSA_3072, RSA_4096
  - Combinaison autorisée de modes d'utilisation clés : {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}

  - REMARQUE : {Encrypt = true, Wrap = true} est la seule option valide lors de l'importation d'une clé publique destinée à chiffrer des données ou à encapsuler une clé
- TR31_S0_CLÉ_ASYMÉTRIQUE_POUR_SIGNATURE NUMÉRIQUE
  - Algorithmes clés autorisés : RSA_2048, RSA_3072, RSA_4096
  - Combinaison autorisée des principaux modes d'utilisation : {Sign = true}, {Verify = true}
  - REMARQUE : {Verify = true} est la seule option valide lors de l'importation d'une clé destinée à la signature, telle qu'un certificat racine, un certificat intermédiaire ou des certificats de signature pour TR-34.
- TR31_K3_KEY_ASYMMETRIC_FOR_KEY_AGREEMENT
  - Utilisé pour les algorithmes d'accord clés tels que l'ECDH
  - Algorithmes clés autorisés : ECC_NIST_P256, ECC_NIST_P384, ECC_NIST_P521
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}.
  - REMARQUE : DeriveKeyUsage est utilisé pour spécifier le type de clé qui sera dérivé de cette clé de base. Ce problème est résolu au moment de la clé creation/import.
- TR31_K2_TR34_CLÉ_ASYMÉTRIQUE
  - Clé asymétrique utilisée pour les mécanismes d'échange de clés X9.24 compatibles tels que TR-34
  - Algorithmes clés autorisés : RSA_2048, RSA_3072, RSA_4096
  - Combinaison autorisée des principaux modes d'utilisation : { DeriveKey = true}.
  - Combinaison autorisée de modes d'utilisation clés : {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}
  - REMARQUE : {Encrypt = true, Wrap = true} est la seule option valide lors de l'importation d'une clé publique destinée à chiffrer des données ou à encapsuler une clé

* Cette combinaison algorithm/key de types n'est actuellement prise en charge par aucune opération cryptographique

Opérations relatives aux données

Une fois que vous avez défini une clé AWS de chiffrement des paiements, celle-ci peut être utilisée pour effectuer des opérations cryptographiques. Les différentes opérations exécutent différents types d'activités, allant du chiffrement au hachage, en passant par des algorithmes spécifiques au domaine tels que la CVV2 génération.

Les données cryptées ne peuvent pas être déchiffrées sans la clé de déchiffrement correspondante (clé symétrique ou clé privée selon le type de chiffrement). De même, les algorithmes de hachage et spécifiques au domaine ne peuvent pas être vérifiés sans la clé symétrique ou la clé publique.

Pour plus d'informations sur les types de clés valides pour des opérations spécifiques, voir Clés valides pour les opérations cryptographiques.

> **Note**
>
> Nous recommandons d'utiliser les données de test dans un environnement hors production. L'utilisation de clés et de données de production (PAN, ID BDK, etc.) dans un environnement hors production peut avoir un impact sur votre périmètre de conformité, par exemple pour les normes PCI DSS et PCI P2PE.

Rubriques

Chiffrer, déchiffrer et rechiffrer les données

Les méthodes de chiffrement et de déchiffrement peuvent être utilisées pour chiffrer ou déchiffrer des données à l'aide de diverses techniques symétriques et asymétriques, notamment TDES, AES et RSA. Ces méthodes prennent également en charge les clés dérivées à l'aide des techniques DUKPT

et EMV. Dans les cas d'utilisation où vous souhaitez sécuriser les données sous une nouvelle clé sans exposer les données sous-jacentes, la ReEncrypt commande peut également être utilisée.

> **Note**
>
> Lorsque vous utilisez les encrypt/decrypt fonctions, toutes les entrées sont supposées être en hexadécimal. Par exemple, une valeur de 1 sera saisie sous la forme 31 (hexadécimal) et un t minuscule est représenté par 74 (hexadécimal). Toutes les sorties sont également en HexBinary.

Pour plus de détails sur toutes les options disponibles, veuillez consulter le guide de l'API pour le chiffrement, le déchiffrement et le rechiffrement.

Rubriques

- Chiffrer des données
- Déchiffrer des données

Chiffrer des données

L'`Encrypt DataAPI` est utilisée pour chiffrer les données à l'aide de clés de chiffrement symétriques et asymétriques ainsi que de clés dérivées du DUKPT et de l'EMV. Différents algorithmes et variantes sont pris en charge`TDES`, notamment, `RSA` et`AES`.

Les entrées principales sont la clé de chiffrement utilisée pour chiffrer les données, les données en texte brut au format HexBinary à chiffrer et les attributs de chiffrement tels que le vecteur d'initialisation et le mode pour les chiffrements par blocs tels que TDES. Les données en texte brut doivent être exprimées en multiples de 8 octets pour`TDES`, 16 octets pour `AES` et de la longueur de la clé dans le cas de`RSA`. Les entrées clés symétriques (TDES, AES, DUKPT, EMV) doivent être complétées dans les cas où les données d'entrée ne répondent pas à ces exigences. Le tableau suivant indique la longueur maximale du texte brut pour chaque type de clé et le type de remplissage que vous définissez `EncryptionAttributes` pour les clés RSA.

Type de rembourrage	RSA_2048	RSA_3072	RSA_4096
`OAEP SHA1`	428	684	940

Type de rembourrage	RSA_2048	RSA_3072	RSA_4096
OAEP SHA256	380	636	892
OAEP SHA512	252	508	764
PKCS1	488	744	1 000
None	488	744	1 000

Les sorties principales incluent les données chiffrées sous forme de texte chiffré au format HexBinary et la valeur de la somme de contrôle pour la clé de chiffrement. Pour plus de détails sur toutes les options disponibles, veuillez consulter le guide de l'API pour Encrypt.

Exemples

- Chiffrer les données à l'aide d'une clé symétrique AES
- Chiffrer les données à l'aide de la clé DUKPT
- Chiffrer les données à l'aide d'une clé symétrique dérivée de l'EMV
- Chiffrer les données à l'aide d'une clé RSA

Chiffrer les données à l'aide d'une clé symétrique AES

> **Note**
>
> Tous les exemples supposent que la clé correspondante existe déjà. Les clés peuvent être créées à l'aide de l'CreateKeyopération ou importées à l'aide de l'ImportKeyopération.

Example

Dans cet exemple, nous chiffrerons les données en texte brut à l'aide d'une clé symétrique créée à l'aide de l'CreateKeyopération ou importée à l'aide de l'opération. ImportKey Pour cette opération, la clé doit être KeyModesOfUse réglée sur `Encrypt` et KeyUsage définie sur`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Consultez la section Clés pour les opérations cryptographiques pour plus d'options.

```
$ aws payment-cryptography-data encrypt-data --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi --plain-text
 313233343132333431323334313233343 --encryption-attributes 'Symmetric={Mode=CBC}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

Chiffrer les données à l'aide de la clé DUKPT

Example

Dans cet exemple, nous allons chiffrer les données en texte brut à l'aide d'une clé DUKPT. AWS Supports de cryptographie des paiements TDES et clés AES DUKPT. Pour cette opération, la clé doit être KeyModesOfUse réglée sur `DeriveKey` et KeyUsage définie sur`TR31_B0_BASE_DERIVATION_KEY`. Consultez la section Clés pour les opérations cryptographiques pour plus d'options.

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --plain-text 31323334313233343132333431323334 --encryption-attributes
 'Dukpt={KeySerialNumber=FFFF9876543210E00001}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

Chiffrer les données à l'aide d'une clé symétrique dérivée de l'EMV

Example

Dans cet exemple, nous allons chiffrer des données en texte clair à l'aide d'une clé symétrique dérivée de l'EMV qui a déjà été créée. Vous pouvez utiliser une commande comme celle-ci pour envoyer des données vers une carte EMV. Pour cette opération, la clé doit être KeyModesOfUse définie sur `Derive` et KeyUsage définie sur `TR31_E1_EMV_MKEY_CONFIDENTIALITY` ou`TR31_E6_EMV_MKEY_OTHER`. Consultez la section Clés pour les opérations cryptographiques pour plus de détails.

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
```

```
 --plain-text 33612AB9D6929C3A828EB6030082B2BD --encryption-attributes
 'Emv={MajorKeyDerivationMode=EMV_OPTION_A,PanSequenceNumber=27,PrimaryAccountNumber=1000000000
 InitializationVector=1500000000000999,Mode=CBC}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

Chiffrer les données à l'aide d'une clé RSA

Example

Dans cet exemple, nous allons chiffrer les données en texte brut à l'aide d'une clé publique RSA importée à l'aide de l'opération. ImportKey Pour cette opération, la clé doit être KeyModesOfUse réglée sur `Encrypt` et KeyUsage définie sur`TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`. Consultez la section Clés pour les opérations cryptographiques pour plus d'options.

Pour le PKCS #7 ou d'autres systèmes de rembourrage non pris en charge actuellement, veuillez en faire la demande avant d'appeler le service et sélectionner aucun rembourrage en omettant l'indicateur de rembourrage « Asymmetric= {} »

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/thfezpmsalcfwmsg
 --plain-text 31323334313233343132333431323334 --encryption-attributes
 'Asymmetric={PaddingType=OAEP_SHA256}'
```

```
{
 "CipherText":
 "12DF6A2F64CC566D124900D68E8AFEAA794CA819876E258564D525001D00AC93047A83FB13 \
 E73F06329A100704FA484A15A49F06A7A2E55A241D276491AA91F6D2D8590C60CDE57A642BC64A897F4832A3930
 \
 0FAEC7981102CA0F7370BFBF757F271EF0BB2516007AB111060A9633D1736A9158042D30C5AE11F8C5473EC70F067
 \
 72590DEA1638E2B41FAE6FB1662258596072B13F8E2F62F5D9FAF92C12BB70F42F2ECDCF56AADF0E311D4118FE3591
 \
 FB672998CCE9D00FFFE05D2CD154E3120C5443C8CF9131C7A6A6C05F5723B8F5C07A4003A5A6173E1B425E2B5E42AD
 \
 7A2966734309387C9938B029AFB20828ACFC6D00CD1539234A4A8D9B94CDD4F23A",
 "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/5dza7xqd6soanjtb",
 "KeyCheckValue": "FF9DE9CE"
}
```

Déchiffrer des données

L'`Decrypt Data`API est utilisée pour déchiffrer les données à l'aide de clés de chiffrement symétriques et asymétriques ainsi que de clés dérivées de DUKPT et EMV. Différents algorithmes et variantes sont pris en charge`TDES`, notamment, `RSA` et`AES`.

Les entrées principales sont la clé de déchiffrement utilisée pour déchiffrer les données, les données chiffrées au format HexBinary à déchiffrer et les attributs de déchiffrement tels que le vecteur d'initialisation, le mode sous forme de blocs, etc. Les sorties principales incluent les données déchiffrées sous forme de texte brut au format HexBinary et la valeur de la somme de contrôle pour la clé de déchiffrement. Pour plus de détails sur toutes les options disponibles, veuillez consulter le guide de l'API pour le déchiffrement.

Exemples

- Déchiffrer les données à l'aide d'une clé symétrique AES
- Déchiffrer les données à l'aide de la clé DUKPT
- Déchiffrer les données à l'aide d'une clé symétrique dérivée d'EMV
- Déchiffrer les données à l'aide d'une clé RSA

Déchiffrer les données à l'aide d'une clé symétrique AES

Example

Dans cet exemple, nous allons déchiffrer les données chiffrées à l'aide d'une clé symétrique. Cet exemple montre une `AES` clé, mais elle `TDES_2KEY` est également prise en charge. `TDES_3KEY` Pour cette opération, la clé doit être KeyModesOfUse réglée sur `Decrypt` et KeyUsage définie sur`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Consultez la section Clés pour les opérations cryptographiques pour plus d'options.

```
$ aws payment-cryptography-data decrypt-data --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi --cipher-text
 33612AB9D6929C3A828EB6030082B2BD --decryption-attributes 'Symmetric={Mode=CBC}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "PlainText": "31323334313233343132333431323334"
}
```

Déchiffrer les données à l'aide de la clé DUKPT

> **Note**
>
> L'utilisation de données de déchiffrement avec DUKPT pour les transactions P2PE peut renvoyer à votre application les données PAN et autres données du titulaire de la carte, qui devront être prises en compte lors de la détermination de son champ d'application PCI DSS.

Example

Dans cet exemple, nous allons déchiffrer les données chiffrées à l'aide d'une clé DUKPT créée à l'aide de l'CreateKeyopération ou importée à l'aide de l'opération. ImportKey Pour cette opération, la clé doit être KeyModesOfUse réglée sur `DeriveKey` et KeyUsage définie sur`TR31_B0_BASE_DERIVATION_KEY`. Consultez la section Clés pour les opérations cryptographiques pour plus d'options. Lorsque vous utilisez`DUKPT`, pour l'`TDES`algorithme, la longueur des données du texte chiffré doit être un multiple de 16 octets. Pour `AES` l'algorithme, la longueur des données du texte chiffré doit être un multiple de 32 octets.

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --cipher-text 33612AB9D6929C3A828EB6030082B2BD --decryption-attributes
 'Dukpt={KeySerialNumber=FFFF9876543210E00001}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "PlainText": "31323334313233343132333431323334"
}
```

Déchiffrer les données à l'aide d'une clé symétrique dérivée d'EMV

Example

Dans cet exemple, nous allons déchiffrer les données de texte chiffré à l'aide d'une clé symétrique dérivée de l'EMV créée à l'aide de l'opération ou importée à l'aide de l'CreateKeyopération. ImportKey Pour cette opération, la clé doit être KeyModesOfUse définie sur `Derive` et KeyUsage définie sur `TR31_E1_EMV_MKEY_CONFIDENTIALITY` ou`TR31_E6_EMV_MKEY_OTHER`. Consultez la section Clés pour les opérations cryptographiques pour plus de détails.

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --cipher-text 33612AB9D6929C3A828EB6030082B2BD --decryption-attributes
 'Emv={MajorKeyDerivationMode=EMV_OPTION_A,PanSequenceNumber=27,PrimaryAccountNumber=1000000000
 InitializationVector=1500000000000999,Mode=CBC}'
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi",
"KeyCheckValue": "71D7AE",
"PlainText": "313233343132333431323334313233343"
}
```

Déchiffrer les données à l'aide d'une clé RSA

Example

Dans cet exemple, nous allons déchiffrer les données de texte chiffré à l'aide d'une paire de clés RSA créée à l'aide de l'opération. CreateKey Pour cette opération, la clé doit être KeyModesOfUse réglée sur Activé `Decrypt` et KeyUsage définie sur`TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`. Consultez la section Clés pour les opérations cryptographiques pour plus d'options.

Pour le PKCS #7 ou d'autres systèmes de rembourrage non pris en charge actuellement, veuillez ne sélectionner aucun rembourrage en omettant l'indicateur de rembourrage « Asymmetric= {} » et supprimez le rembourrage après avoir appelé le service.

```
$ aws payment-cryptography-data decrypt-data \
       --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5dza7xqd6soanjtb --cipher-text
 8F4C1CAFE7A5DEF9A40BEDE7F2A264635C... \
       --decryption-attributes 'Asymmetric={PaddingType=OAEP_SHA256}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/5dza7xqd6soanjtb",
    "KeyCheckValue": "FF9DE9CE",
    "PlainText": "31323334313233343132333431323334"
}
```

Génération et vérification des données de carte

La génération et la vérification des données de carte intègrent des données dérivées des données de carte, par exemple CVV CVV2, CVC et DCVV.

Rubriques

- Générer des données de carte
- Vérifier les données de la carte

Générer des données de carte

L'`Generate Card Data`API est utilisée pour générer des données de carte à l'aide d'algorithmes tels que CVV CVV2 ou Dynamic CVV2. Pour savoir quelles clés peuvent être utilisées pour cette commande, consultez la section Clés valides pour les opérations cryptographiques.

De nombreuses valeurs cryptographiques telles que CVV, iCVV CVV2, CAVV V7 utilisent le même algorithme cryptographique mais font varier les valeurs d'entrée. Par exemple, CardVerificationValue1 contient les entrées « numéro de ServiceCode carte » et « date d'expiration ». Bien que CardVerificationValue2 ne possède que deux de ces entrées, cela est dû au fait que pour CVV2/CVC2, le ServiceCode est fixé à 000. De même, pour iCVV, ServiceCode il est fixé à 999. Certains algorithmes peuvent réutiliser les champs existants, tels que CAVV V8, auquel cas vous devrez consulter le manuel de votre fournisseur pour obtenir les valeurs d'entrée correctes.

> **Note**
>
> La date d'expiration doit être saisie dans le même format (tel que MMYY ou YYMM) pour la génération et la validation afin de produire des résultats corrects.

Générer CVV2

Example

Dans cet exemple, nous allons générer un CVV2 pour un PAN donné avec des entrées indiquant la date d'expiration de la carte PAN et la date d'expiration de la carte. Cela suppose qu'une clé de vérification de carte a été générée.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue2={CardExpiryDate=0123}
```

```
  {
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
      "KeyCheckValue": "CADDA1",
      "ValidationData": "801"
  }
```

Générer iCVV

Example

Dans cet exemple, nous allons générer un iCVV pour un PAN donné avec les entrées suivantes : un code de PAN service 999 et une date d'expiration de la carte. Cela suppose qu'une clé de vérification de carte a été générée.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1",
    "ValidationData": "801"
}
```

Vérifier les données de la carte

`Verify Card Data`est utilisé pour vérifier les données créées à l'aide d'algorithmes de paiement basés sur des principes de cryptage tels que`DISCOVER_DYNAMIC_CARD_VERIFICATION_CODE`.

Les valeurs d'entrée sont généralement fournies dans le cadre d'une transaction entrante à un émetteur ou à un partenaire de plateforme de support. Pour vérifier un cryptogramme ARQC (utilisé pour les cartes à puce EMV), veuillez consulter Vérifier l'ARQC.

Pour plus d'informations, consultez VerifyCardValidationDatale guide de l'API.

Si la valeur est vérifiée, l'API renverra http/200. Si la valeur n'est pas vérifiée, elle renverra http/400.

Vérifiez CVV2

Example

Dans cet exemple, nous allons valider un CVV/ CVV2 pour un PAN donné. CVV2 Il est généralement fourni par le titulaire de la carte ou l'utilisateur au moment de la transaction pour validation. Afin de valider leur saisie, les valeurs suivantes seront fournies lors de l'exécution : clé à utiliser pour la validation (CVK)PAN, date d'expiration de la carte et CVV2 saisies. Le format d'expiration de la carte doit correspondre à celui utilisé lors de la génération de valeur initiale.

Pour tous les paramètres disponibles, voir CardVerificationValue2 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue2={CardExpiryDate=0123} --validation-data 801
```

```
  {
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
      "KeyCheckValue": "CADDA1"
  }
```

Vérifiez iCVV

Example

Dans cet exemple, nous allons vérifier un iCVV pour un PAN donné en saisissant la clé à utiliser pour la validation (CVK), le code de service 999PAN, la date d'expiration de la carte et l'iCVV fourni par la transaction à valider.

iCVV n'est pas une valeur saisie par l'utilisateur (comme CVV2) mais intégrée sur une carte EMV. Il convient de déterminer s'il doit toujours être validé lorsqu'il est fourni.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999} --validation-data 801
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1",
    "ValidationData": "801"
}
```

Générez, traduisez et vérifiez les données PIN

Les fonctions de données PIN vous permettent de générer des codes PIN aléatoires, des valeurs de vérification des codes PIN (PVV) et de valider les codes PIN chiffrés entrants par rapport au PVV ou aux décalages de code PIN.

La traduction par épingle vous permet de traduire une épingle d'une touche fonctionnelle à une autre sans exposer l'épingle en texte clair, conformément à l'exigence 1 du code PIN PCI.

> **Note**
>
> Étant donné que la génération et la validation du code PIN sont généralement des fonctions de l'émetteur et que la traduction du code PIN est une fonction typique de l'acquéreur, nous

vous recommandons de prendre en compte l'accès le moins privilégié et de définir des politiques adaptées au cas d'utilisation de vos systèmes.

Rubriques

Translate les données du code PIN

Les fonctions Translate PIN data sont utilisées pour traduire les données PIN cryptées d'un jeu de clés à un autre sans que les données chiffrées ne quittent le HSM. Il est utilisé pour le chiffrement P2PE où les clés de travail doivent changer mais où le système de traitement n'a pas besoin de déchiffrer les données ou n'est pas autorisé à le faire. Les entrées principales sont les données cryptées, la clé de chiffrement utilisée pour chiffrer les données, les paramètres utilisés pour générer les valeurs d'entrée. L'autre ensemble d'entrées est constitué des paramètres de sortie demandés, tels que la clé à utiliser pour chiffrer la sortie et les paramètres utilisés pour créer cette sortie. Les principaux résultats sont un ensemble de données nouvellement crypté ainsi que les paramètres utilisés pour le générer.

> **Note**
>
> Pour garantir la conformité à la norme PCI, les PrimaryAccountNumber valeurs entrantes et sortantes doivent correspondre. La traduction d'un code PIN d'un PAN à un autre n'est pas autorisée.

Rubriques

Code PIN de PEK à DUKPT

Example

Dans cet exemple, nous allons convertir un code PIN d'un bloc PIN AES ISO 4 utilisant le chiffrement DUKPT en un code PEK TDES utilisant un bloc PIN ISO 0. Cela est courant lorsqu'un terminal de paiement chiffre un code PIN en ISO 4, puis il peut être retraduit en TDES pour un traitement en aval si la prochaine connexion ne prend pas encore en charge l'AES.

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-block
 "AC17DC148BDA645E" --outgoing-translation-
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}' --outgoing-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt  --incoming-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/4pmyquwjs3yj4vwe --incoming-translation-attributes
 IsoFormat4="{PrimaryAccountNumber=171234567890123}" --incoming-dukpt-attributes
 KeySerialNumber="FFFF9876543210E00008"
```

```
    {
            "PinBlock": "1F4209C670E49F83E75CC72E81B787D9",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
            "KeyCheckValue": "7CC9E2"
        }
```

Code PIN de PEK à PEK

Example

Dans cet exemple, nous traduisons un code PIN chiffré sous un PEK (clé de cryptage PIN) en un autre PEK. Ceci est couramment utilisé lors du routage des transactions entre différents systèmes ou partenaires qui utilisent des clés de chiffrement différentes, tout en maintenant la conformité au code PIN PCI en le cryptant tout au long du processus. Les deux clés utilisent le chiffrement TDES 3KEY dans cet exemple, mais diverses options sont disponibles, notamment AES ISO-4 à TDES ISO-0, DUKPT à PEK ou à PEK. AS2805

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-block
 "AC17DC148BDA645E" \
    --incoming-translation-attributes
 IsoFormat0='{PrimaryAccountNumber=171234567890123}' \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --outgoing-translation-attributes
 IsoFormat0='{PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh
```

```
{
    "PinBlock": "E8F2A6C4D1B93E7F",
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
    "KeyCheckValue": "9A325B"
}
```

Le bloc PIN de sortie est désormais crypté sous le deuxième PEK et peut être transmis en toute sécurité au système en aval qui détient la clé correspondante.

Générer des données PIN

Les fonctions de génération de données PIN sont utilisées pour générer des valeurs liées au code PIN, telles que le PVV et les décalages par blocs de code utilisés pour valider la saisie du code PIN par les utilisateurs pendant la transaction ou le délai d'autorisation. Cette API peut également générer une nouvelle épingle aléatoire à l'aide de divers algorithmes.

Générez un code PIN aléatoire et un visa PVV correspondant

Example

Dans cet exemple, nous allons générer une nouvelle épingle (aléatoire) dont les sorties seront chiffrées `PIN block` (PinData. PinBlock) et a PVV (PinData.offset). Les entrées principales sont les PAN suivantes : le`Pin Verification Key`, le `Pin Encryption Key` et le`PIN block format`.

Cette commande nécessite que la clé soit de type`TR31_V2_VISA_PIN_VERIFICATION_KEY`.

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-
attributes VisaPin={PinVerificationKeyIndex=1}
```

```
{
            "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "GenerationKeyCheckValue": "7F2363",
            "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
            "EncryptionKeyCheckValue": "7CC9E2",
            "EncryptedPinBlock": "AC17DC148BDA645E",
            "PinData": {
                "VerificationValue": "5507"
            }
        }
```

Générez un visa PVV pour un code PIN connu

Example

Dans cet exemple, nous allons générer un PVV pour un code PIN (crypté) donné. Un code PIN crypté peut être reçu en amont, par exemple en provenance d'un terminal de paiement ou d'un titulaire de carte, en utilisant le flux de code PIN sélectionnable par l'utilisateur. Les entrées principales sont les PAN suivantes : le`Pin Verification Key`, le`Pin Encryption Key`, le `Encrypted Pin Block` et le`PIN block format`.

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt --primary-account-number
 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-attributes
 VisaPinVerificationValue={PinVerificationKeyIndex=1,EncryptedPinBlock=AA584CED31790F37}
```

```
{
            "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "GenerationKeyCheckValue": "7F2363",
            "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
            "EncryptionKeyCheckValue": "7CC9E2",
            "EncryptedPinBlock": "AC17DC148BDA645E",
            "PinData": {
                "VerificationValue": "5507"
            }
        }
```

Générer un décalage d' IBM3624 épingle pour une épingle

IBM 3624 PIN Offset est aussi parfois appelée méthode IBM. Cette méthode génère un natural/intermediate code PIN à l'aide des données de validation (généralement le PAN) et d'une clé PIN (PVK). Les épingles naturelles sont en fait une valeur dérivée et le caractère déterministe est très efficace à gérer pour un émetteur, car aucune donnée PIN n'a besoin d'être stockée au niveau du titulaire de la carte. L'inconvénient le plus évident est que ce système ne prend pas en compte les épingles sélectionnables ou aléatoires par le titulaire de la carte. Pour autoriser ces types de broches, un algorithme de décalage a été ajouté au schéma. Le décalage représente la différence entre l'épingle sélectionnée (ou aléatoire) par l'utilisateur et la touche naturelle. La valeur de décalage est

enregistrée par l'émetteur ou le processeur de la carte. Au moment de la transaction, le service AWS de cryptographie des paiements recalcule en interne le code PIN naturel et applique le décalage pour trouver le code PIN. Il compare ensuite cette valeur à la valeur fournie par l'autorisation de transaction.

Plusieurs options existent pour IBM3624 :

- `Ibm3624NaturalPin`affichera le code PIN naturel et un bloc de code PIN crypté
- `Ibm3624PinFromOffset`générera un bloc PIN crypté en fonction d'un décalage
- `Ibm3624RandomPin`générera un code PIN aléatoire, puis le décalage correspondant et le bloc de code crypté.
- `Ibm3624PinOffset`génère le décalage de broche en fonction d'une épingle sélectionnée par l'utilisateur.

En interne à la cryptographie des AWS paiements, les étapes suivantes sont effectuées :

- Ajoutez le plan fourni à 16 caractères. Si <16 est indiqué, tapez sur le côté droit en utilisant le caractère de rembourrage fourni.
- Chiffre les données de validation à l'aide de la clé de génération du code PIN.
- Décimalisez les données chiffrées à l'aide de la table de décimalisation. Cela fait correspondre des chiffres hexadécimaux à des chiffres décimaux, par exemple « A » peut correspondre à 9 et 1 peut correspondre à 1.
- Obtenez les 4 premiers chiffres à partir d'une représentation hexadécimale de la sortie. C'est l'épingle naturelle.
- Si un code PIN a été sélectionné par l'utilisateur ou généré au hasard, modulo soustrait le code naturel du code PIN du client. Le résultat est le décalage de la broche.

Exemples

- Exemple : générer un décalage d' IBM3624 épingle pour une épingle

Exemple : générer un décalage d' IBM3624 épingle pour une épingle

Dans cet exemple, nous allons générer une nouvelle épingle (aléatoire) dont les sorties seront chiffrées `PIN block` (PinData. PinBlock) et une valeur de `IBM3624` décalage (Pindata.offset). Les

entrées sont PAN les données de validation (généralement le pan), le caractère de remplissage, lePin Verification Key, le Pin Encryption Key et lePIN block format.

Cette commande nécessite que la clé de génération du code PIN soit de type TR31_V1_IBM3624_PIN_VERIFICATION_KEY et que la clé de chiffrement soit de type TR31_P0_PIN_ENCRYPTION_KEY

Example

L'exemple suivant montre la génération d'une épingle aléatoire, puis la sortie du bloc de code PIN crypté et de la valeur de IBM3624 décalage à l'aide de Ibm3624. RandomPin

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt --primary-account-number
 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-attributes
 Ibm3624RandomPin="{DecimalizationTable=9876543210654321,PinValidationDataPadCharacter=D,PinVal
```

```
{
              "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
              "GenerationKeyCheckValue": "7F2363",
              "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
              "EncryptionKeyCheckValue": "7CC9E2",
              "EncryptedPinBlock": "AC17DC148BDA645E",
              "PinData": {
                  "PinOffset": "5507"
              }
          }
```

Vérifier les données du code PIN

Les fonctions de vérification des données PIN sont utilisées pour vérifier si un code PIN est correct. Cela implique généralement de comparer la valeur du code PIN précédemment enregistrée à celle saisie par le titulaire de la carte lors d'un POI. Ces fonctions comparent deux valeurs sans exposer la valeur sous-jacente de l'une ou l'autre des sources.

Valider le code PIN crypté en utilisant la méthode PVV

Example

Dans cet exemple, nous allons valider un code PIN pour un PAN donné. Le code PIN est généralement fourni par le titulaire de la carte ou l'utilisateur au moment de la transaction à des fins de validation et est comparé à la valeur enregistrée (l'entrée du titulaire de la carte est fournie sous forme de valeur cryptée par le terminal ou un autre fournisseur en amont). Afin de valider cette entrée, les valeurs suivantes seront également fournies lors de l'exécution : la clé utilisée pour chiffrer le code PIN d'entrée (il s'agit souvent d'un`IWK`) PAN et la valeur par rapport à laquelle vérifier (un `PVV` ou`PIN offset`).

Si AWS Payment Cryptography est en mesure de valider le code PIN, un http/200 est renvoyé. Si le code PIN n'est pas validé, il renverra un http/400.

```
$ aws payment-cryptography-data verify-pin-data --verification-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0  --
verification-attributes VisaPin="{PinVerificationKeyIndex=1,VerificationValue=5507}" --
encrypted-pin-block AC17DC148BDA645E
```

```
        {
            "VerificationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "VerificationKeyCheckValue": "7F2363",
            "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
            "EncryptionKeyCheckValue": "7CC9E2",

        }
```

Valider le code PIN crypté à l'aide de la méthode PVV - erreur : code PIN incorrect

Example

Dans cet exemple, nous allons essayer de valider un code PIN pour un PAN donné, mais cela échouera car le code PIN est incorrect.

Lors de l'utilisation SDKs, cela apparaît sous la forme « {" Message » « Échec de la vérification du bloc d'épingles ». , « Motif » « INVALID_PIN »}

```
$ aws payment-cryptography-data verify-pin-data --verification-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0  --
verification-attributes VisaPin="{PinVerificationKeyIndex=1,VerificationValue=9999}" --
encrypted-pin-block AC17DC148BDA645E
```

```
       An error occurred (VerificationFailedException) when calling the VerifyPinData
 operation: Pin block verification failed.
```

Valider le code PIN crypté à l'aide de la méthode PVV - erreur, mauvaises entrées

Example

Dans cet exemple, nous allons essayer de valider un code PIN pour un PAN donné, mais cela échouera en raison de mauvaises entrées et du fait que les données entrantes n'étaient pas un code PIN valide. Les causes courantes sont les suivantes : 1/mauvaise touche utilisée 2/les paramètres d'entrée tels que le format du bloc panoramique ou pin sont incorrects 3/le bloc pin est endommagé.

```
$ aws payment-cryptography-data verify-pin-data --verification-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier  --primary-account-number 171234567890123
 --pin-block-format ISO_FORMAT_0  --verification-attributes
 VisaPin="{PinVerificationKeyIndex=1,VerificationValue=9999}" --encrypted-pin-block
 AC17DC148BDA645E
```

```
       An error occurred (ValidationException) when calling the VerifyPinData
operation: Pin block provided is invalid. Please check your input to ensure all field
values are correct.
```

Valider un code PIN par rapport au décalage de IBM3624 code précédemment enregistré

Dans cet exemple, nous validerons le code PIN fourni par le titulaire de la carte par rapport à l'offset du code enregistré auprès de l'émetteur/du processeur de la carte. Les entrées sont similaires ??? à l'ajout du code PIN crypté fourni par le terminal de paiement (ou un autre fournisseur en amont tel que le réseau de cartes). Si le code PIN correspond, l'API renverra http 200, où les sorties seront cryptées `PIN block` (PinData. PinBlock) et une valeur de `IBM3624` décalage (Pindata.offset).

Cette commande nécessite que la clé de génération du code PIN soit de type `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` et que la clé de chiffrement soit de type `TR31_P0_PIN_ENCRYPTION_KEY`

Example

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt --primary-account-number
 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-attributes
 Ibm3624RandomPin="{DecimalizationTable=9876543210654321,PinValidationDataPadCharacter=D,PinVal
```

```
{
"GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
"GenerationKeyCheckValue": "7F2363",
"EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
"EncryptionKeyCheckValue": "7CC9E2",
"EncryptedPinBlock": "AC17DC148BDA645E",
"PinData": {
  "PinOffset": "5507"
}
}
```

Cryptogramme de demande d'authentification (ARQC)

L'API de cryptogramme de demande d'authentification de vérification est utilisée pour vérifier l'ARQC. La génération de l'ARQC n'entre pas dans le cadre de la cryptographie des AWS paiements et est généralement effectuée sur une carte à puce EMV (ou un équivalent numérique tel qu'un portefeuille mobile) pendant le temps d'autorisation de la transaction. Un ARQC est unique à chaque transaction et est destiné à montrer de manière cryptographique à la fois la validité de la carte et à garantir que les données de transaction correspondent exactement à la transaction en cours (attendue).

AWS La cryptographie des paiements fournit diverses options pour valider l'ARQC et générer des valeurs ARPC facultatives, notamment celles définies dans le livre 2 de l'EMV 4.4 et d'autres schémas utilisés par Visa et Mastercard. Pour une liste complète de toutes les options disponibles, consultez la VerifyCardValidationData section du guide de l'API.

Les cryptogrammes ARQC nécessitent généralement les entrées suivantes (bien que cela puisse varier en fonction de l'implémentation) :

- PAN - Spécifié dans le PrimaryAccountNumber champ

- Numéro de séquence PAN (PSN) - spécifié dans le champ PanSequenceNumber
- Méthode de dérivation des clés, telle que la clé de session commune (CSK), spécifiée dans le SessionKeyDerivationAttributes
- Mode de dérivation de la clé principale (tel que l'option A EMV) - Spécifié dans le MajorKeyDerivationMode
- Données de transaction - une chaîne de diverses données de transaction, de terminal et de carte telles que le montant et la date - spécifiées dans le TransactionData champ
- Clé principale de l'émetteur : clé principale utilisée pour dériver la clé cryptographique (AC) utilisée pour protéger les transactions individuelles et spécifiée dans le champ KeyIdentifier

Rubriques

Création de données de transaction

Le contenu exact (et l'ordre) du champ de données de transaction varie en fonction de l'implémentation et du schéma de réseau, mais les champs minimaux recommandés (et la séquence de concaténation) sont définis dans le livre 2 de l'EMV 4.4, section 8.1.1 - Sélection des données. Si les trois premiers champs sont le montant (17,00), l'autre montant (0,00) et le pays d'achat, les données de transaction commenceront comme suit :

- 000000001700 - montant - 12 positions décimales implicites à deux chiffres
- 000000000000 - autre montant - 12 positions décimales implicites à deux chiffres
- 0124 - code de pays à quatre chiffres
- Données de transaction (partielles) de sortie - 00000000170000000000000000124

Rembourrage des données de transaction

Les données de transaction doivent être complétées avant d'être envoyées au service. La plupart des schémas utilisent le remplissage de la méthode 2 ISO 9797, où une chaîne hexadécimale est ajoutée par 80 puis par 00 jusqu'à ce que le champ soit un multiple de la taille du bloc de chiffrement ; 8

octets ou 16 caractères pour TDES et 16 octets ou 32 caractères pour AES. L'alternative (méthode 1) n'est pas aussi courante mais utilise uniquement 00 comme caractères de remplissage.

Rembourrage ISO 9797 Méthode 1

Non rembourré :
0000000000000000000000084000800080000840160517000000000938000 0B03011203 (74 caractères ou 37 octets)

Rembourré :
0000000000000000000000084000800080000840160517000000000938000 0B03011203 000000 (80 caractères ou 40 octets)

Rembourrage ISO 9797 Méthode 2

Non rembourré :
0000000000000000000000084000800080000840160517000000000938000 0B1F220103000000 (80 caractères ou 40 octets)

Rembourré :
0000000000000000000000084000800080000840160517000000000938000 0B1F220103000000 8000000000000000 (88 caractères ou 44 octets)

Exemples

Visa CVN10

Example

Dans cet exemple, nous allons valider un ARQC généré à l'aide de Visa CVN10.

Si AWS Payment Cryptography est en mesure de valider l'ARQC, un http/200 est renvoyé. Si l'ARCQ (Authorization Request Cryptogram) n'est pas validé, il renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram D791093C8A921769 \
--key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
--major-key-derivation-mode EMV_OPTION_A \
--transaction-data
 0000000017000000000000000840008000800008401605170000000009380000B03011203000000 \
--session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
,"PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
  "KeyCheckValue": "08D7B4"
}
```

Visa CVN18 et visa CVN22

Example

Dans cet exemple, nous allons valider un ARQC généré à l'aide de Visa CVN18 ou CVN22. Les opérations cryptographiques sont les mêmes entre CVN18 et CVN22 mais les données contenues dans les données de transaction varient. Par rapport à CVN10, un cryptogramme complètement différent est généré même avec les mêmes entrées.

Si AWS Payment Cryptography est en mesure de valider l'ARQC, un http/200 est renvoyé. Si l'ARCQ n'est pas validé, il renverra un http/400.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram \
--auth-request-cryptogram 61EDCC708B4C97B4
--key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
--major-key-derivation-mode EMV_OPTION_A
--transaction-data
 00000000170000000000000084000800080000840160517000000000093800000B1F22010300000000000
 \
00000000000000000000000000000000000000000800000000000000
--session-key-derivation-attributes='{"EmvCommon":
{"ApplicationTransactionCounter":"000B", \
"PanSequenceNumber":"01","PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
  "KeyCheckValue": "08D7B4"
}
```

Générer et vérifier MAC

Les codes d'authentification de message (MAC) sont généralement utilisés pour authentifier l'intégrité d'un message (s'il a été modifié). Les hachages cryptographiques tels que le HMAC (code d'authentification des Hash-Based messages) CBC-MAC et le CMAC (code d'authentification des Cipher-based messages) fournissent une assurance supplémentaire à l'expéditeur du MAC en utilisant la cryptographie. HMAC est basé sur des fonctions de hachage tandis que CMAC est basé sur des chiffrements par blocs. Le service prend également en charge les algorithmes ISO9797 1 et 3 qui sont des types de. CBC-MACs

Tous les algorithmes MAC de ce service combinent une fonction de hachage cryptographique et une clé secrète partagée. Ils prennent un message et une clé secrète, tels que le contenu clé d'une clé, et renvoient un tag ou un mac unique. Si un seul caractère du message change, ou si la clé secrète change, le tag obtenu est totalement différent. En exigeant une clé secrète, les MAC cryptographiques garantissent également l'authenticité ; il est impossible de générer un mac identique sans la clé secrète. Les MAC cryptographiques sont parfois appelés signatures symétriques, car ils fonctionnent comme des signatures numériques, mais utilisent une clé unique pour la signature et la vérification.

AWS La cryptographie des paiements prend en charge plusieurs types de MAC :

ALGORITHME ISO9797 1

Désigné par `KeyUsage` ISO9797_ALGORITHM1. Si le champ n'est pas un multiple de la taille du bloc (8 caractères bytes/16 hexadécimaux pour TDES, 16 bytes/32 caractères pour AES), AWS Payment Cryptography applique automatiquement la méthode de remplissage 1 ISO9797. Si d'autres méthodes de rembourrage sont nécessaires, vous pouvez les appliquer avant d'appeler le service.

ALGORITHME ISO9797 3 (MAC de détail)

Désigné par `KeyUsage` ISO9797_ALGORITHM3. Les mêmes règles de remplissage s'appliquent que celles de l'algorithme 1

ALGORITHME ISO9797 5 (CMAC)

Désigné par ou `KeyUsage` TR31_M6_ISO_9797_5_CMAC_KEY

HMAC

Désigné par `KeyUsage` TR31_M7_HMAC_KEY, y compris HMAC_SHA224, HMAC_SHA256, HMAC_SHA384 et HMAC_SHA512

AS2805.4.1 MAC

Désigné par `KeyUsage` TR31_M0_ISO_16609_MAC_KEY. Pour plus de détails sur l'AS2805, voir ???

DUKPT MAC

DUKPT MAC est généralement utilisé pour confirmer la source et la charge utile des messages to/from des terminaux de paiement. Il déduit une clé à l'aide des techniques de dérivation

DUKPT, puis exécute le MAC. Les clés utilisées avec cette option sont désignées par `KeyUsage` TR31_B0_BASE_DERIVATION_KEY.

EMV MAC

Le MAC EMV est généralement appelé clé d'intégrité dans la documentation EMV. Il déduit une clé à l'aide des techniques de dérivation EMV, puis utilise ISO9797_ALGORITHM3 en interne. Il est généralement utilisé pour envoyer des scripts d'émetteur à une carte à puce à des fins de reprogrammation. Les clés utilisées avec cette option sont désignées par `KeyUsage` TR31_E2_EMV_MKEY_INTEGRITY. Si vous envoyez un script et mettez à jour un code PIN hors ligne, veillez à GenerateMacEmvPinChangece qu'il effectue ces deux opérations.

Rubriques

- Générer un MAC
- Vérifiez le MAC

Générer un MAC

L'API Generate MAC est utilisée pour authentifier les données relatives aux cartes, telles que le suivi des données provenant d'une bande magnétique de carte, en utilisant des clés cryptographiques connues pour générer un code d'authentification de message (MAC) pour la validation des données entre les parties émettrices et réceptrices. Les données utilisées pour générer le MAC incluent des données de message, une clé de cryptage MAC secrète et un algorithme MAC pour générer une valeur MAC unique pour la transmission. Le destinataire du MAC utilisera les mêmes données de message MAC, la même clé de chiffrement MAC et le même algorithme pour reproduire une autre valeur MAC à des fins de comparaison et d'authentification des données. Même si un caractère du message change ou si la clé MAC utilisée pour la vérification n'est pas identique, la valeur MAC obtenue est différente. L'API prend en charge les clés de chiffrement MAC ISO 9797-1 Algorithme 1 et ISO 9797-1 Algorithme 3 MAC (utilisant une clé MAC statique et une clé DUKPT dérivée), HMAC et EMV MAC pour cette opération.

La valeur d'entrée pour `message-data` doit être des données HexBinary.

Pour plus d'informations sur toutes les options de cette API, consultez GenerateMacet VerifyMac.

Le paramètre optionnel mac-length vous permet de tronquer la valeur de sortie (bien que cela puisse également être fait dans votre code). Une longueur de 8 correspond à 8 octets ou 16 caractères hexadécimaux.

Les clés MAC peuvent être créées avec AWS Payment Cryptography en appelant CreateKeyou importées en appelant ImportKey.

> **Note**
>
> Les algorithmes CMAC et HMAC ne nécessitent pas de rembourrage. Tous les autres nécessitent que les données soient complétées en fonction de la taille de bloc de l'algorithme, qui est un multiple de 8 octets (16 caractères hexadécimaux) pour le TDES et de 16 octets (32 caractères hexadécimaux) pour l'AES.

Exemples

- Générer un HMAC
- Générer un MAC à l'aide de l'algorithme ISO 9797-1 3
- Générer un MAC à l'aide de CMAC
- Générer un MAC à l'aide de DUKPT CMAC

Générer un HMAC

Dans cet exemple, nous allons générer un code HMAC (Hash-Based Message Authentication Code) pour l'authentification des données de carte à l'aide de l'algorithme HMAC `HMAC_SHA256` et de la clé de cryptage HMAC. La clé doit être KeyUsage réglée sur `TR31_M7_HMAC_KEY` et KeyModesOfUse sur`Generate`. La longueur de hachage (par exemple 256) est définie lors de la création de la clé et ne peut pas être modifiée.

Le paramètre optionnel mac-length réduira le MAC de sortie, bien que cela puisse également être effectué en dehors du service. Cette valeur est en octets, donc une valeur de 16 suppose une chaîne hexadécimale de 32 caractères.

Example

```
$ aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6 \
    --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" \
    --generation-attributes Algorithm=HMAC
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "KeyCheckValue": "2976E7",
    "Mac": "ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C"
}
```

Générer un MAC à l'aide de l'algorithme ISO 9797-1 3

Dans cet exemple, nous allons générer un MAC à l'aide de l'algorithme ISO 9797-1 3 (Retail MAC) pour l'authentification des données de carte. La clé doit être KeyUsage réglée sur `TR31_M3_ISO_9797_3_MAC_KEY` et KeyModesOfUse sur`Generate`.

Example

```
$ aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h \
    --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" \
    --generation-attributes="Algorithm=ISO9797_ALGORITHM3"
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
    "KeyCheckValue": "2976EA",
    "Mac": "A8F7A73DAF87B6D0"
}
```

Générer un MAC à l'aide de CMAC

Le CMAC est le plus souvent utilisé lorsque les touches sont en AES, mais il prend également en charge le TDES. Dans cet exemple, nous allons générer un MAC à l'aide du CMAC (algorithme ISO 9797-1 5) pour l'authentification des données de carte à l'aide d'une clé AES. La clé doit être KeyUsage réglée sur `TR31_M6_ISO_9797_5_CMAC_KEY` et KeyModesOfUse sur`Generate`.

Example

```
$ aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" \
    --generation-attributes Algorithm="CMAC"
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "C1EB8F",
    "Mac": "1F8C36E63F91E4E93DF7842BF5E2E5F7"
}
```

Générer un MAC à l'aide de DUKPT CMAC

Cet exemple génère un MAC en utilisant DUKPT (Derived Unique Key Per Transaction) avec CMAC pour l'authentification des données de carte. La clé doit être KeyUsage définie sur `TR31_B0_BASE_DERIVATION_KEY` et KeyModesOfUse `DeriveKey` définie sur true.

Les clés DUKPT dérivent une clé unique pour chaque transaction à l'aide d'une clé de dérivation de base (BDK) et d'un numéro de série de clé (KSN).

Example

```
$ aws payment-cryptography-data generate-mac --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/qnobl5lghrzunce6 --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" --generation-
attributes="DukptCmac={KeySerialNumber="932A6E954ABB32DD00000001",DukptKeyVariant=BIDIRECTIONAL
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "KeyCheckValue": "C1EB8F",
    "Mac": "1F8C36E63F91E4E93DF7842BF5E2E5F7"
}
```

Vérifiez le MAC

L'API Verify MAC est utilisée pour vérifier le MAC (code d'authentification des messages) pour l'authentification des données liées à la carte. Il doit utiliser la même clé de chiffrement utilisée lors de la génération du MAC pour reproduire la valeur MAC à des fins d'authentification. La clé de chiffrement MAC peut être créée avec AWS Payment Cryptography en appelant CreateKeyou importée par appel ImportKey. L'API prend en charge les clés de chiffrement DUKPT MAC, HMAC et EMV MAC pour cette opération.

Si la valeur est vérifiée, le paramètre de réponse `MacDataVerificationSuccessful` sera renvoyé`Http/200`, sinon `Http/400` avec un message l'indiquant`Mac verification failed`.

Exemples

- Vérifiez HMAC
- Vérifiez le MAC à l'aide de DUKPT CMAC

Vérifiez HMAC

Dans cet exemple, nous allons vérifier un code HMAC (Hash-Based Message Authentication Code) pour l'authentification des données de carte à l'aide de l'algorithme HMAC `HMAC_SHA256` et de la clé de cryptage HMAC. La clé doit être KeyUsage définie sur `TR31_M7_HMAC_KEY` et KeyModesOfUse `Verify` définie sur true.

Example

```
$ aws payment-cryptography-data verify-mac \
     --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6 \
     --message-data
 "3b343038383439303031303733393431353d323430383232363030303730303033f33" \
     --mac ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C \
     --verification-attributes Algorithm=HMAC_SHA256
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "KeyCheckValue": "2976E7"
}
```

Vérifiez le MAC à l'aide de DUKPT CMAC

Dans cet exemple, nous allons vérifier un MAC à l'aide du DUKPT (Derived Unique Key Per Transaction) avec le CMAC pour l'authentification des données de carte. La clé doit être KeyUsage définie sur `TR31_B0_BASE_DERIVATION_KEY` et KeyModesOfUse `DeriveKey` définie sur true. Les clés DUKPT dérivent une clé unique pour chaque transaction à l'aide d'une clé de dérivation de base (BDK) et d'un numéro de série de clé (KSN). La valeur de DukptKeyVariant doit correspondre entre l'expéditeur et le destinataire. REQUEST est généralement utilisé d'un terminal à l'autre, VERIFY d'un backend à un autre et BIDIRECTIONAL lorsqu'une seule touche est utilisée dans les deux sens.

Example

```
$ aws payment-cryptography-data verify-mac \
     --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
     --message-data
 "3b343038383439303031303733393431353d323430383232363030303730303033f33" \
     --mac D8E804EE74BF1D909A2C01C0BDE8EF34 \
     --verification-attributes
 DukptCmac='{"KeySerialNumber":"932A6E954ABB32DD00000001","DukptKeyVariant":"BIDIRECTIONAL"}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "C1EB8F"
}
```

Clés valides pour les opérations cryptographiques

Certaines touches ne peuvent être utilisées que pour certaines opérations. En outre, certaines opérations peuvent limiter les modes d'utilisation des touches. Consultez le tableau suivant pour connaître les combinaisons autorisées.

> **Note**
>
> Certaines combinaisons, bien qu'autorisées, peuvent créer des situations inutilisables, telles que la génération de codes CVV (`generate`) mais l'impossibilité de les vérifier. (`verify`)

Rubriques

- GenerateCardData
- VerifyCardData
- GeneratePinData (pour les VISA/ABA programmes)
- GeneratePinData (pourIBM3624)
- VerifyPinData (pour les VISA/ABA programmes)
- VerifyPinData (pourIBM3624)

- Déchiffrer des données
- Encrypt Data
- Translate Pin Data
- Générer/vérifier un MAC
- GenerateMacEmvPinChange
- VerifyAuthRequestCryptogram
- Clé d'Import/Export
- Types de clés non utilisés

GenerateCardData

Point de terminaison API	Opération ou algorithme cryptographique	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
GenerateCardData	• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	{Générer = vrai}, {Générer = vrai, Vérifier = vrai}
GenerateCardData	• VALEUR_VÉRIFICATION_1 DE LA CARTE_1 • VALEUR_VÉRIFICATION_2 DE LA CARTE_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY	{Générer = vrai}, {Générer = vrai, Vérifier = vrai}

Point de terminaison API	Opération ou algorithme cryptographique	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
GenerateCardData	• VALEUR_AUTHENTIFICATION_VÉRIFICATION_DU_TITULAIRE DE LA CARTE	TR31_E6_EMV_MKEY_AUTRE	• TDES_2KEY	{ DeriveKey = vrai}
GenerateCardData	• CODE DE VÉRIFICATION DE LA CARTE DYNAMIQUE	TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS	• TDES_2KEY	{ DeriveKey = vrai}
GenerateCardData	• VALEUR_VÉRIFICATION_DYNAMIQUE DE LA CARTE	TR31_E6_EMV_MKEY_AUTRE	• TDES_2KEY	{ DeriveKey = vrai}

VerifyCardData

Opération ou algorithme cryptographique	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
• AMEX_CARD_SECURITY	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	{Générer = vrai}, {Générer = vrai, Vérifier = vrai}

Opération ou algorithme cryptographique	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2			
• VALEUR_VÉRIFICATION_1 DE LA CARTE_1 • VALEUR_VÉRIFICATION_2 DE LA CARTE_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY	{Générer = vrai}, {Générer = vrai, Vérifier = vrai}
• VALEUR_AUTHENTIFICATION_VÉRIFICATION_DU_TITULAIRE DE LA CARTE	TR31_E6_EMV_MKEY_AUTRE	• TDES_2KEY	{ DeriveKey = vrai}
• CODE DE VÉRIFICATION DE LA CARTE DYNAMIQUE	TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS	• TDES_2KEY	{ DeriveKey = vrai}
• VALEUR_VÉRIFICATION_DYNAMIQUE DE LA CARTE	TR31_E6_EMV_MKEY_AUTRE	• TDES_2KEY	{ DeriveKey = vrai}

GeneratePinData (pour les VISA/ABA programmes)

`VISA_PIN or VISA_PIN_VERIFICATION_VALUE`

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé de cryptage PIN	TR31CLÉ DE CHIFFREMENT _P0_PIN	• TDES_2KEY • TDES_3KEY	• {Chiffrer = vrai, envelopper = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Clé de génération de code PIN	TR31_V2_V ISA_PIN_VERIFICATI ON_KEY	• TDES_3KEY	• {Générer = vrai} • {Générer = vrai, vérifier = vrai}

GeneratePinData (pour**IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé de cryptage PIN	TR31CLÉ DE CHIFFREMENT _P0_PIN	• TDES_2KEY • TDES_3KEY	Pour IBM3624 _NATURAL_PIN, _RANDOM_PIN,

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
			IBM3624 _PIN_FROM _OFFSET IBM3624 • {Chiffrer = vrai, envelopper = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai} Pour IBM3624 _PIN_OFFSET • {Chiffrer = vrai, Décompresser = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Clé de génération de code PIN	TR31_V1_ _CLÉ IBM3624 DE VÉRIFICATION PAR CODE PIN	• TDES_3KEY	• {Générer = vrai} • {Générer = vrai, vérifier = vrai}

VerifyPinData (pour les VISA/ABA programmes)

`VISA_PIN`

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé de cryptage PIN	TR31CLÉ DE CHIFFREMENT _P0_PIN	• TDES_2KEY • TDES_3KEY	• {Déchiffrer = vrai, Décompresser = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Clé de génération de code PIN	TR31_V2_V ISA_PIN_VERIFICATI ON_KEY	• TDES_3KEY	• {Vérifier = vrai} • {Générer = vrai, vérifier = vrai}

VerifyPinData (pour**IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé de cryptage PIN	TR31CLÉ DE CHIFFREMENT _P0_PIN	• TDES_2KEY • TDES_3KEY	Pour IBM3624 _NATURAL_PIN, _RANDOM_PIN,

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
			IBM3624 _PIN_FROM _OFFSET IBM3624 • {Déchiffrer = vrai, Décompresser = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Clé de vérification du code PIN	TR31_V1_ _CLÉ IBM3624 DE VÉRIFICATION PAR CODE PIN	• TDES_3KEY	• {Vérifier = vrai} • {Générer = vrai, vérifier = vrai}

Déchiffrer des données

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
DUKPT	TR31_B0_B ASE_DERIV ATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = vrai} • { NoRestrictions = vrai}

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
EMV	TR31_E1_EMV_MKEY_CONFIDENTIALITÉ TR31_E6_EMV_MKEY_AUTRE	• TDES_2KEY	• { DeriveKey = vrai}
RSA	TR31_D1_CLÉ ASYMÉTRIQUE POUR LE CHIFFREMENT DES DONNÉES	• RSA_2048 • RSA_3072 • RSA_4096	• {Déchiffrer = vrai, unwrap=vrai} • {Encrypt=True, Wrap=True, Decrypt = true, Unwrap=True}
Clés symétriques	TR31_D0_CLÉ_DE CHIFFREMENT DES DONNÉES SYMÉTRIQUES	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Déchiffrer = vrai, unwrap=vrai} • {Encrypt=True, Wrap=True, Decrypt = true, Unwrap=True} • { NoRestrictions = vrai}

Encrypt Data

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = vrai} • { NoRestrictions = vrai}
EMV	TR31_E1_EMV_MKEY_CONFIDENTIALITÉ TR31_E6_EMV_MKEY_AUTRE	• TDES_2KEY	• { DeriveKey = vrai}
RSA	TR31_D1_CLÉ ASYMÉTRIQUE POUR LE CHIFFREMENT DES DONNÉES	• RSA_2048 • RSA_3072 • RSA_4096	• {Chiffrer = vrai, wrap=vrai} • {Encrypt=True, Wrap=True, Decrypt = true, Unwrap=True}
Clés symétriques	TR31_D0_CLÉ_DE CHIFFREMENT DES DONNÉES SYMÉTRIQUES	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Chiffrer = vrai, wrap=vrai} • {Encrypt=True, Wrap=True, Decrypt = true, Unwrap=True} • { NoRestrictions = vrai}

Translate Pin Data

Direction	Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Source de données entrante	DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = vrai} • { NoRestrictions = vrai}
Source de données entrante	Non dopé (PEK, AWK, IWK, etc.)	TR31CLÉ DE CHIFFREMENT_P0_PIN	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Déchiffrer = vrai, Décompresser = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Cible de données sortantes	DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = vrai} • { NoRestrictions = vrai}
Cible de données sortantes	Non dopé (PEK, IWK, AWK, etc.)	TR31CLÉ DE CHIFFREMENT_P0_PIN	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192	• {Chiffrer = vrai, envelopper = vrai}

Direction	Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
			• AES_256	• {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}

Générer/vérifier un MAC

Les clés MAC sont utilisées pour créer des hachages cryptographiques message/body d'une donnée. Il n'est pas recommandé de créer une clé avec des modes d'utilisation limités car vous ne pourrez pas effectuer l'opération correspondante. Cependant, vous pouvez import/export utiliser une touche avec une seule opération si l'autre système est destiné à effectuer l'autre moitié de la paire d'opérations.

Utilisation autorisée des clés	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé MAC	TR31_M1_ISO_9797_1_MAC_KEY	• TDES_2KEY • TDES_3KEY	• {Générer = vrai} • {Générer = vrai, vérifier = vrai} • {Vérifier = vrai} • {Générer = vrai}

Utilisation autorisée des clés	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé MAC (MAC pour le commerce de détail)	TR31_M1_ISO_9797_3_MAC_KEY	• TDES_2KEY • TDES_3KEY	• {Générer = vrai} • {Générer = vrai, vérifier = vrai} • {Vérifier = vrai} • {Générer = vrai}
Clé MAC (CMAC)	TR31_M6_ISO_9797_5_CMAC_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Générer = vrai} • {Générer = vrai, vérifier = vrai} • {Vérifier = vrai} • {Générer = vrai}
Clé MAC (HMAC)	TR31_M7_HMAC_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Générer = vrai} • {Générer = vrai, vérifier = vrai} • {Vérifier = vrai}
Clé MAC (AS2805)	TR31_M0_ISO_16609_MAC_KEY	• TDES_2KEY • TDES_3KEY	• {Générer = vrai} • {Générer = vrai, vérifier = vrai} • {Vérifier = vrai}

GenerateMacEmvPinChange

GenerateMacEmvPinChange combine la génération de MAC et le chiffrement du code PIN pour les opérations de changement de code PIN hors ligne EMV. Cette opération nécessite deux types de clés différents : une clé d'intégrité pour la génération de MAC et une clé de confidentialité pour le chiffrement par code PIN.

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé d'intégrité de la messagerie sécurisée	TR31_E2_EMV_MKEY_INTEGRITY	• TDES_2KEY	• { NoRestrictions = vrai}
Clé de confidentialité de la messagerie sécurisée	TR31_E1_EMV_MKEY_CONFIDENTIALITÉ	• TDES_2KEY	• { DeriveKey = vrai}
PIN PEK actuel (clé de cryptage PIN)	TR31CLÉ DE CHIFFREMENT _P0_PIN	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Déchiffrer = vrai, Décompresser = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Nouveau code PIN PEK (clé de cryptage PIN)	TR31CLÉ DE CHIFFREMENT _P0_PIN	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Déchiffrer = vrai, Décompresser = vrai} • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai} • { NoRestrictions = vrai}
Clé ARQC	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMMES	• TDES_2KEY	• { DeriveKey = vrai}

Type de clé	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Note S'applique uniquement aux systèmes de dérivation Visa et Amex.			

VerifyAuthRequestCryptogram

Utilisation autorisée des clés	Option EMV	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
• OPTION A • OPTION B	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMMES	• TDES_2KEY	• { DeriveKey = vrai}

Clé d'Import/Export

Type d'opération	Utilisation autorisée des clés	Algorithme clé autorisé	Combinaison autorisée des principaux modes d'utilisation
Clé d'emballage TR-31	TR31_K1_KEY_BLOCK_PROTECTION_KEY TR31_K0_KEY_CRYPTION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Encrypt = true, Wrap = true} (exportation uniquement) • {Decrypt = true, Unwrap = true} (importation uniquement) • {Chiffrer = vrai, Déchiffrer = vrai, envelopper = vrai, déballer = vrai}
Importation d'une autorité de certification approuvée	TR31_S0_CLÉ_ASYMÉTRIQUE POUR SIGNATURE NUMÉRIQUE	• RSA_2048 • RSA_3072 • RSA_4096	• {Vérifier = vrai}
Importation d'un certificat à clé publique pour le chiffrement asymétrique	TR31_D1_CLÉ ASYMÉTRIQUE POUR LE CHIFFREMENT DES DONNÉES	• RSA_2048 • RSA_3072 • RSA_4096	• {Encrypt=TRUE, WRAP=TRUE}
Clé utilisée pour les algorithmes d'accord clés tels que l'ECDH	TR31_K3_CLÉ_ASYMÉTRIQUE_POUR_KEY_AGREEMENT	• ECC_NIST_P256 • ECC_NIST_P384 • ECC_NIST_P521	• { DeriveKey = vrai}

Types de clés non utilisés

Les types de clés suivants ne sont pas actuellement utilisés par AWS Payment Cryptography :

- TR31_P1_PIN_GENERATION_KEY

Cas d'utilisation courants

AWS La cryptographie des paiements prend en charge de nombreuses opérations cryptographiques de paiement classiques. Les rubriques suivantes servent de guide sur l'utilisation de ces opérations pour les cas d'utilisation courants typiques. Pour obtenir la liste de toutes les commandes, veuillez consulter l'API AWS Payment Cryptography.

Rubriques

Émetteurs et processeurs d'émetteurs

Les cas d'utilisation par les émetteurs se composent généralement de quelques parties. Cette section est organisée par fonction (utilisation d'épingles, par exemple). Dans un système de production, les clés sont généralement limitées à un compartiment de cartes donné et sont créées lors de la configuration du compartiment plutôt qu'en ligne, comme indiqué ici.

Rubriques

Fonctions générales

Rubriques

Générez une épingle aléatoire et le PVV associé, puis vérifiez la valeur

Rubriques

Créez la ou les clés

Pour générer un code PIN aléatoire et le code PVV, vous aurez besoin de deux clés : une clé de vérification du code PIN (PVK) pour générer le code PIN et une clé de cryptage du code PIN pour chiffrer le code PIN. Le code PIN lui-même est généré de manière aléatoire en toute sécurité dans le service et n'est lié cryptographiquement à aucune des clés.

Le PGK doit être une clé de l'algorithme TDES_2KEY basé sur l'algorithme PVV lui-même. Un PEK peut être TDES_2KEY, TDES_3KEY ou AES_128. Dans ce cas, étant donné que le PEK est destiné à un usage interne au sein de votre système, AES_128 serait un bon choix. Si un PEK est utilisé pour échanger avec d'autres systèmes (par exemple, des réseaux de cartes, des acquéreurs ATMs) ou s'il est déplacé dans le cadre d'une migration, TDES_2KEY peut être le choix le plus approprié pour des raisons de compatibilité.

Créez le PEK

```
$ aws payment-cryptography create-key \
               --exportable
               --key-attributes
 KeyAlgorithm=AES_128,KeyUsage=TR31_P0_PIN_ENCRYPTION_KEY,\
                   KeyClass=SYMMETRIC_KEY,\
                   KeyModesOfUse='{Encrypt=true,Decrypt=true,Wrap=true,Unwrap=true}' --
tags='[{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
               "Key": {
                   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
                   "KeyAttributes": {
```

```
                    "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "AES_128",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "7CC9E2",
                "KeyCheckValueAlgorithm": "CMAC",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/ivi5ksfsuplneuyt. Vous en aurez besoin à l'étape suivante.

Créez le PVK

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_V2_VISA_PIN_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyMo
  --tags='[{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza",
```

```
                "KeyAttributes": {
                    "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "51A200",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/ov6icy4ryas4zcza. Vous en aurez besoin à l'étape suivante.

Générez un code PIN aléatoire, générez le PVV et renvoyez le code PIN et le PVV cryptés

Example

Dans cet exemple, nous allons générer un nouveau code PIN (aléatoire) à 4 chiffres dont les sorties seront chiffrées `PIN block` (PinData. PinBlock) et a `PVV` (PinData. VerificationValue). Les entrées principales sont `PAN` le format `Pin Verification Key` (également connu sous le nom de clé de génération de code PIN) `Pin Encryption Key` et le format PIN Block.

Cette commande nécessite que la clé soit de type`TR31_V2_VISA_PIN_VERIFICATION_KEY`.

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
```

```
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-
attributes VisaPin={PinVerificationKeyIndex=1}
```

```
{
            "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "GenerationKeyCheckValue": "7F2363",
            "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
            "EncryptionKeyCheckValue": "7CC9E2",
            "EncryptedPinBlock": "AC17DC148BDA645E",
            "PinData": {
                "VerificationValue": "5507"
            }
        }
```

Valider le code PIN crypté en utilisant la méthode PVV

Example

Dans cet exemple, nous allons valider un code PIN pour un PAN donné. Le code PIN est généralement fourni par le titulaire de la carte ou l'utilisateur au moment de la transaction à des fins de validation et est comparé à la valeur enregistrée (l'entrée du titulaire de la carte est fournie sous forme de valeur cryptée par le terminal ou un autre fournisseur en amont). Afin de valider cette entrée, les valeurs suivantes seront également fournies lors de l'exécution : le code PIN crypté, la clé utilisée pour chiffrer le code PIN d'entrée (souvent appelé IWK) PAN et la valeur par rapport à laquelle vérifier (un PVV ou`PIN offset`).

Si AWS Payment Cryptography parvient à valider le code PIN, un http/200 est renvoyé. Si le code PIN n'est pas validé, il renverra un http/400.

```
$ aws payment-cryptography-data verify-pin-data --verification-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0  --
verification-attributes VisaPin="{PinVerificationKeyIndex=1,VerificationValue=5507}" --
encrypted-pin-block AC17DC148BDA645E
```

```
{
```

```
        "VerificationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
        "VerificationKeyCheckValue": "7F2363",
        "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
        "EncryptionKeyCheckValue": "7CC9E2",
}
```

Générer ou vérifier un CVV pour une carte donnée

Le CVV ou CVV1 est une valeur traditionnellement intégrée à la bande magnétique d'une carte. Ce n'est pas la même chose que CVV2 (visible pour le titulaire de la carte et pour les achats en ligne).

La première étape consiste à créer une clé. Pour ce didacticiel, vous allez créer une clé CVK 3DES double longueur (2KEY TDES).

> **Note**
>
> CVV CVV2 et iCVV utilisent tous des algorithmes similaires, voire identiques, mais font varier les données d'entrée. Ils utilisent tous le même type de clé TR31 _C0_CARD_VERIFICATION_KEY, mais il est recommandé d'utiliser des clés distinctes pour chaque utilisation. Ils peuvent être distingués à l'aide de and/or balises alias, comme dans l'exemple ci-dessous.

Créez la clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVV"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
r52o3wbqxyf6qlqr",
                "KeyAttributes": {
```

```
                "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
                "KeyClass": "SYMMETRIC_KEY",
                "KeyAlgorithm": "TDES_2KEY",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": true,
                    "Sign": false,
                    "Verify": true,
                    "DeriveKey": false,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "DE89F9",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
            "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
        }
    }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/r52o3wbqxyf6qlqr. Vous en aurez besoin à l'étape suivante.

Générer un CVV

Example

Dans cet exemple, nous allons générer un CVV pour un PAN donné avec les entrées suivantes : 121 comme code de PAN service (tel que défini par ISO/IEC 7813) et date d'expiration de la carte.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
r52o3wbqxyf6qlqr --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=121}'
```

```
                {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/r52o3wbqxyf6qlqr",
                    "KeyCheckValue": "DE89F9",
                    "ValidationData": "801"
                }
```

Valider le CVV

Example

Dans cet exemple, nous allons vérifier un CVV pour un PAN donné en saisissant un CVK, un code de service de 121PAN, la date d'expiration de la carte et le CVV fourni lors de la transaction à valider.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

> **Note**
>
> Le CVV n'est pas une valeur saisie par l'utilisateur (comme CVV2) mais il est généralement intégré à une bande magnétique. Il convient de déterminer s'il doit toujours être validé lorsqu'il est fourni.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/r52o3wbqxyf6qlqr
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=121} --validation-data 801
```

```
{
                    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
r52o3wbqxyf6qlqr",
                    "KeyCheckValue": "DE89F9",
                    "ValidationData": "801"
}
```

Générer ou vérifier une CVV2 pour une carte spécifique

CVV2est une valeur qui est traditionnellement indiquée au dos d'une carte et qui est utilisée pour les achats en ligne. Pour les cartes virtuelles, elles peuvent également être affichées sur une application ou un écran. Sur le plan cryptographique, c'est le même que CVV1 , mais avec une valeur de code de service différente.

Créez la clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVV2"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
               "Key": {
                   "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
                   "KeyAttributes": {
                       "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
                       "KeyClass": "SYMMETRIC_KEY",
                       "KeyAlgorithm": "TDES_2KEY",
                       "KeyModesOfUse": {
                           "Encrypt": false,
                           "Decrypt": false,
                           "Wrap": false,
                           "Unwrap": false,
                           "Generate": true,
                           "Sign": false,
                           "Verify": true,
                           "DeriveKey": false,
                           "NoRestrictions": false
                       }
                   },
                   "KeyCheckValue": "AEA5CD",
                   "KeyCheckValueAlgorithm": "ANSI_X9_24",
                   "Enabled": true,
                   "Exportable": true,
                   "KeyState": "CREATE_COMPLETE",
                   "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
```

```
                "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/7f7g4spf3xcklhzu. Vous en aurez besoin à l'étape suivante.

Générez un CVV2

Example

Dans cet exemple, nous allons générer un CVV2pour un PAN donné avec des entrées indiquant la date d'expiration de la carte PAN et la date d'expiration de la carte.

Pour tous les paramètres disponibles, voir CardVerificationValue2 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/7f7g4spf3xcklhzu
 --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue2='{CardExpiryDate=1127}'
```

```
                    {
                        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
                        "KeyCheckValue": "AEA5CD",
                        "ValidationData": "321"
                    }
```

Valider un CVV2

Example

Dans cet exemple, nous allons vérifier un CVV2pour un PAN donné en saisissant un CVK, la date d'expiration de la carte PAN et le CVV fournis lors de la transaction à valider.

Pour tous les paramètres disponibles, voir CardVerificationValue2 dans le guide de référence de l'API.

> **Note**
>
> CVV2 et les autres entrées sont des valeurs saisies par l'utilisateur. Ce n'est donc pas nécessairement le signe d'un problème que cela échoue périodiquement à valider.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/7f7g4spf3xcklhzu
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue2='{CardExpiryDate=1127} --validation-data 321
```

```
{
                      "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
                      "KeyCheckValue": "AEA5CD",
                      "ValidationData": "801"
   }
```

Générer ou vérifier un iCVV pour une carte spécifique

iCVV utilise le même algorithme que CVV/ CVV2 mais iCVV est intégré dans une carte à puce. Son code de service est 999.

Créez la clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"ICVV"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
               "Key": {
                   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
c7dsi763r6s7lfp3",
                   "KeyAttributes": {
```

```
                    "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "1201FB",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/c7dsi763r6s7lfp3. Vous en aurez besoin à l'étape suivante.

Générer un iCVV

Example

Dans cet exemple, nous allons générer un iCVV pour un PAN donné avec les entrées suivantes : un code de PAN service (tel que défini par ISO/IEC 7813) de 999 et la date d'expiration de la carte.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
c7dsi763r6s7lfp3 --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999}'
```

```
                    {
                        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/c7dsi763r6s7lfp3",
                        "KeyCheckValue": "1201FB",
                        "ValidationData": "532"
                    }
```

Valider iCVV

Example

Pour la validation, les entrées sont le CVKPAN, un code de service 999, la date d'expiration de la carte et l'iCVV fourni lors de la transaction à valider.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

> **Note**
>
> iCVV n'est pas une valeur saisie par l'utilisateur (comme CVV2) mais est généralement intégrée à une EMV/chip carte. Il convient de déterminer s'il doit toujours être validé lorsqu'il est fourni.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/c7dsi763r6s7lfp3
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999} --validation-data 532
```

```
{
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/c7dsi763r6s7lfp3",
                    "KeyCheckValue": "1201FB",
                    "ValidationData": "532"
```

```
    }
```

Vérifiez un ARQC EMV et générez un ARPC

L'ARQC (Authorization Request Cryptogram) est un cryptogramme généré par une carte EMV (puce) et utilisé pour valider les détails de la transaction ainsi que l'utilisation d'une carte autorisée. Il intègre les données de la carte, du terminal et de la transaction elle-même.

Au moment de la validation sur le backend, les mêmes entrées sont fournies à AWS Payment Cryptography, le cryptogramme est recréé en interne et celui-ci est comparé à la valeur fournie avec la transaction. En ce sens, il est similaire à un MAC. Le livre 2 de l'EMV 4.4 définit trois aspects de cette fonction : les méthodes de dérivation de clés (connues sous le nom de clé de session commune - CSK) pour générer des clés de transaction uniques, une charge utile minimale et les méthodes de génération d'une réponse (ARPC).

Les schémas de cartes individuels peuvent spécifier des champs transactionnels supplémentaires à intégrer ou l'ordre dans lequel ces champs apparaissent. D'autres schémas de dérivation spécifiques au schéma (généralement obsolètes) existent également et sont abordés ailleurs dans cette documentation.

Pour plus d'informations, consultez VerifyCardValidationDatale guide de l'API.

Créez la clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN18"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                    "KeyAttributes": {
                        "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                        "KeyClass": "SYMMETRIC_KEY",
                        "KeyAlgorithm": "TDES_2KEY",
                        "KeyModesOfUse": {
```

```
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": false,
                        "Sign": false,
                        "Verify": false,
                        "DeriveKey": true,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "08D7B4",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/pw3s6nl62t5ushfk. Vous en aurez besoin à l'étape suivante.

Générer un ARQC

L'ARQC est généré exclusivement par une carte EMV. En tant que telle, la cryptographie des AWS paiements ne permet pas de générer une telle charge utile. À des fins de test, un certain nombre de bibliothèques sont disponibles en ligne et peuvent générer une charge utile appropriée ainsi que des valeurs connues généralement fournies par les différents schémas.

Valider un ARQC

Example

Si AWS Payment Cryptography est en mesure de valider l'ARQC, un http/200 est renvoyé. Un ARPC (réponse) peut éventuellement être fourni et inclus dans la réponse une fois l'ARQC validé.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram
 --auth-request-cryptogram 61EDCC708B4C97B4 --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk
```

```
 --major-key-derivation-mode EMV_OPTION_A --transaction-data
 0000000017000000000000000840008000800008401605170000000093800000B1F220103000000000000000000
 --session-key-derivation-attributes='{"EmvCommon":
{"ApplicationTransactionCounter":"000B",
 "PanSequenceNumber":"01","PrimaryAccountNumber":"9137631040001422"}}' --auth-response-
attributes='{"ArpcMethod2":{"CardStatusUpdate":"12345678"}}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
    "KeyCheckValue": "08D7B4",
    "AuthResponseValue":"2263AC85"
}
```

Générer et vérifier un MAC EMV

EMV MAC est un MAC utilisant l'entrée d'une clé dérivée d'EMV, puis exécutant un MAC ISO9797 -3 (Retail) sur les données résultantes. Le MAC EMV est généralement utilisé pour envoyer des commandes à une carte EMV, telles que des scripts de déblocage.

> **Note**
>
> AWS La cryptographie des paiements ne valide pas le contenu du script. Veuillez consulter le manuel de votre schéma ou de votre carte pour plus de détails sur les commandes spécifiques à inclure.

Pour plus d'informations, consultez MacAlgorithmEmvle guide de l'API.

Rubriques

- Créez la clé
- Générer un MAC EMV

Créez la clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E2_EMV_MKEY_INTEGRITY,KeyClass=SYMMETRIC_KEY,KeyModesOfUs
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN18"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).


```
{
        "Key": {
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
            "KeyAttributes": {
                "KeyUsage": "TR31_E2_EMV_MKEY_INTEGRITY",
                "KeyClass": "SYMMETRIC_KEY",
                "KeyAlgorithm": "TDES_2KEY",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "08D7B4",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
            "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
        }
    }
```


Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/pw3s6nl62t5ushfk. Vous en aurez besoin à l'étape suivante.

Générer un MAC EMV

Généralement, un processus principal génère un script EMV (tel que le déblocage d'une carte), le signe à l'aide de cette commande (qui déduit une clé à usage unique spécifique à une carte en particulier), puis renvoie le MAC. Ensuite, la commande + MAC est envoyée à la carte à appliquer.

L'envoi de la commande à la carte ne relève pas du champ d'application de la cryptographie des AWS paiements.

> **Note**
>
> Cette commande est destinée aux commandes lorsqu'aucune donnée cryptée (telle que le code PIN) n'est envoyée. EMV Encrypt peut être combiné à cette commande pour ajouter des données chiffrées au script de l'émetteur avant d'appeler cette commande

Données du message

Les données du message incluent l'en-tête et la commande APDU. Bien que cela puisse varier selon l'implémentation, cet exemple est l'en-tête APDU pour le déblocage (84 24 00 00 08), suivi de l'ATC (0007) puis de l'ARQC de la transaction précédente (999E57 F47CACE). FD0 Le service ne valide pas le contenu de ce champ.

Mode de dérivation des clés de session

Ce champ définit le mode de génération de la clé de session. EMV_COMMON_SESSION_KEY est généralement utilisé pour les nouvelles implémentations, tandis que EMV2000 | AMEX | MASTERCARD_SESSION_KEY | VISA peut également être utilisé.

MajorKeyDerivationMode

EMV définit le mode A, B ou C. Le mode A est le plus courant et la cryptographie des AWS paiements prend actuellement en charge le mode A ou le mode B.

PAN

Le numéro de compte, généralement disponible dans le champ de puce 5A ou ISO8583 le champ 2, mais peut également être extrait du système de carte.

PSN

Le numéro de séquence de la carte. S'il n'est pas utilisé, entrez 00.

SessionKeyDerivationValue

Il s'agit des données de dérivation par session. Il peut s'agir du dernier ARQC (ApplicationCryptogram) du champ 9F26 ou du dernier ATC du champ 9F36 selon le schéma de dérivation.

Remplissage

Le rembourrage est automatiquement appliqué et utilise la méthode de rembourrage ISO/IEC 9797-1 2.

Example

```
$ aws payment-cryptography-data generate-mac --message-data
 8424000008000799E57FD0F47CACE --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk --message-
data 8424000008999E57FD0F47CACE0007 --generation-attributes
 EmvMac="{MajorKeyDerivationMode=EMV_OPTION_A,PanSequenceNumber='00',PrimaryAccountNumber='2235
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
"KeyCheckValue": "08D7B4",
"Mac":"5652EEDF83EA0D84"
}
```

Générer un MAC EMV pour le changement de code PIN

Le changement de code PIN EMV combine deux opérations : générer un code MAC pour un script émetteur et chiffrer un nouveau code PIN pour un changement de code PIN hors ligne sur une carte à puce EMV. Cette commande n'est nécessaire que dans certains pays où le code PIN est enregistré sur la carte à puce (c'est courant dans les pays européens). Ceci est couramment utilisé lorsqu'un titulaire de carte doit changer son code PIN et que le nouveau code PIN doit être transmis de manière sécurisée à la carte avec un MAC pour vérifier l'authenticité de la commande.

> **Note**
>
> Si vous devez uniquement envoyer des commandes à la carte sans modifier le code PIN, pensez plutôt à utiliser les commandes ARPC CSU ou Generate EMV MAC.

Pour plus d'informations, consultez GenerateMacEmvPinChangele guide de l'API.

Générez un MAC EMV et un code PIN crypté pour le changement de code PIN

Cette opération nécessite deux clés : une clé d'intégrité EMV (KeyUsage: TR31 _E2_EMV_MKEY_INTEGRITY) pour la génération de MAC et une clé de confidentialité EMV

(: _E4_EMV_MKEY_CONFIDENTIALITY) pour le chiffrement par code PIN. KeyUsage TR31 Généralement, un processus principal génère un script de modification du code PIN EMV, qui inclut à la fois le MAC du script émetteur et le nouveau code PIN crypté. La commande et le code PIN chiffré sont ensuite envoyés à la carte pour mettre à jour le code PIN hors ligne. L'envoi de la commande à la carte ne relève pas du champ d'application de la cryptographie des AWS paiements.

Données du message

Les données du message incluent la commande APDU pour le script de l'émetteur. Le service ne valide pas le contenu de ce champ.

Nouveau bloc PIN crypté

Le nouveau bloc PIN crypté qui sera envoyé à la carte. Cela doit être fourni sous forme de valeur cryptée à l'aide d'une clé de cryptage par code PIN.

Nouvel identifiant PIN PEK

La clé utilisée pour chiffrer le nouveau code PIN avant qu'il ne soit transmis à cette API.

Clé d'intégrité de la messagerie sécurisée

La clé d'intégrité EMV (KeyUsage: TR31 _E2_EMV_MKEY_INTEGRITY) utilisée pour la génération de MAC.

Clé de confidentialité de la messagerie sécurisée

La clé de confidentialité EMV (KeyUsage: TR31 _E4_EMV_MKEY_CONFIDENTIALITY) utilisée pour le chiffrement par code PIN.

MajorKeyDerivationMode

EMV définit le mode A, B ou C. Le mode A est le plus courant et la cryptographie des AWS paiements prend actuellement en charge le mode A ou le mode B.

Mode

Le mode de cryptage, généralement CBC pour les opérations de changement de code PIN.

PAN

Le numéro de compte, généralement disponible dans le champ de puce 5A ou ISO8583 le champ 2, mais peut également être extrait du système de carte.

PanSequenceNumber

Le numéro de séquence de la carte. S'il n'est pas utilisé, entrez 00.

ApplicationCryptogram

Il s'agit des données de dérivation par session, généralement le dernier ARQC du champ 9F26.

PinBlockLengthPosition

Spécifie l'endroit où la longueur du bloc PIN est codée. Généralement défini sur NONE. Vérifiez les spécifications de votre système de cartes en cas de doute.

PinBlockPaddingType

Spécifie le type de rembourrage pour le bloc PIN. Généralement défini sur NO_PADDING. Vérifiez les spécifications de votre système de cartes en cas de doute.

Example

```
$ aws payment-cryptography-data generate-mac-emv-pin-change \
    --message-data 00A4040008A000000004101080D8050000000101 0A04000000000000 \
    --new-encrypted-pin-block 67FB27C75580EFE7 \
    --new-pin-pek-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --pin-block-format ISO_FORMAT_0 \
    --secure-messaging-confidentiality-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi \
    --secure-messaging-integrity-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/pw3s6nl62t5ushfk \
    --derivation-method-attributes
 'EmvCommon={ApplicationCryptogram=1234567890123457,MajorKeyDerivationMode=EMV_OPTION_A,Mode=CB
```

```
{
    "SecureMessagingIntegrityKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/pw3s6nl62t5ushfk",
    "SecureMessagingIntegrityKeyCheckValue": "08D7B4",
    "SecureMessagingConfidentialityKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi",
    "SecureMessagingConfidentialityKeyCheckValue": "C1EB8F",
    "Mac": "5652EEDF83EA0D84",
    "EncryptedPinBlock": "F1A2B3C4D5E6F7A8"
}
```

Fonctions spécifiques au réseau

Rubriques

Fonctions spécifiques à Visa

Rubriques

ARQC -/ CVN18CVN22

CVN18 et CVN22 utilisez la méthode CSK de dérivation des clés. Les données de transaction exactes varient entre ces deux méthodes. Consultez la documentation du schéma pour plus de détails sur la création du champ de données de transaction.

ARQC - CVN10

CVN10 est une ancienne méthode Visa pour les transactions EMV qui utilise la dérivation de clé par carte plutôt que la dérivation de session (par transaction) et utilise également une charge utile différente. Pour plus d'informations sur le contenu de la charge utile, veuillez contacter le programme pour plus de détails.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN10"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
```

```
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": false,
                        "Sign": false,
                        "Verify": false,
                        "DeriveKey": true,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "08D7B4",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/pw3s6nl62t5ushfk. Vous en aurez besoin à l'étape suivante.

Valider l'ARQC

Example

Dans cet exemple, nous allons valider un ARQC généré à l'aide de Visa CVN10.

Si AWS Payment Cryptography est en mesure de valider l'ARQC, un http/200 est renvoyé. Si l'arqc n'est pas validé, il renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram D791093C8A921769 \
```

```
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
  --major-key-derivation-mode EMV_OPTION_A \
  --transaction-data
 0000000017000000000000000840008000800008401605170000000009380000B03011203000000 \
  --session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
  ,"PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
    "KeyCheckValue": "08D7B4"
  }
```

3DS CAVV V7

Pour les transactions Visa Secure (3DS), une valeur de vérification de l'authentification du titulaire de la carte (CAVV) est générée par le serveur de contrôle d'accès (ACS) de l'émetteur. Le CAVV prouve que l'authentification du titulaire de la carte a eu lieu. Il est unique pour chaque transaction d'authentification et est fourni par l'acquéreur dans le message d'autorisation. CAVV v7 lie des données supplémentaires sur la transaction à l'approbation, y compris des éléments tels que le nom du commerçant, le montant de l'achat et la date d'achat. De cette manière, il s'agit effectivement d'un hachage cryptographique de la charge utile de la transaction.

Sur le plan cryptographique, le CAVV V7 utilise l'algorithme CVV, mais les entrées ont toutes été de la changed/repurposed. Please consult appropriate third party/Visa documentation sur la façon de produire les entrées pour générer une charge utile CAVV V7.

Créez la clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CAVV-V7"},
{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
```

```
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
                "KeyAttributes": {
                    "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "F3FB13",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/dnaeyrjgdjjtw6dk. Vous en aurez besoin à l'étape suivante.

Générer un CAVV V7

Example

Dans cet exemple, nous allons générer un CAVV V7 pour une transaction donnée avec des entrées telles que spécifiées dans les spécifications. Notez que pour cet algorithme, les champs peuvent être réutilisés/réutilisés. Il ne faut donc pas supposer que les étiquettes des champs correspondent aux entrées.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=431}'
```

```
            {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
                "KeyCheckValue": "F3FB13",
                "ValidationData": "491"
            }
```

Valider CAVV V7

Example

Pour la validation, les entrées sont le CVK, les valeurs d'entrée calculées et le CAVV fourni lors de la transaction à valider.

Pour tous les paramètres disponibles, voir CardVerificationValue1 dans le guide de référence de l'API.

> **Note**
>
> Le CAVV n'est pas une valeur saisie par l'utilisateur (similaire CVV2) mais est calculé par l'émetteur ACS. Il convient de se demander s'il doit toujours être validé lorsqu'il est fourni.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/dnaeyrjgdjjtw6dk
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=431} --validation-data 491
```

```
{
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
```

```
                "KeyCheckValue": "F3FB13",
                "ValidationData": "491"
}
```

DCVV (valeur de vérification dynamique de la carte) - CVN17

Le DCVV (Dynamic Card Verification Value) est un cryptogramme dynamique spécifique à Visa utilisé pour les transactions EMV sans contact. Il est connu sous le nom d'EMV et offre une sécurité renforcée en générant une valeur de vérification unique pour chaque transaction. Le DCVV utilise des entrées telles que le numéro de compte principal (PAN), le numéro de séquence PAN (PSN), le compteur de transactions d'application (ATC), le nombre imprévisible et les données de suivi. Il est toujours utilisé à certains endroits, mais a été principalement remplacé par d'autres algorithmes tels que CVN18.

Pour tous les paramètres disponibles, consultez DynamicCardVerificationValuele guide de référence de l'API.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"DCVV"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
```

```
                    "Verify": true,
                    "DeriveKey": false,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "A8E4D2",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2025-02-02T11:45:30.648000-08:00",
            "UsageStartTimestamp": "2025-02-02T11:45:30.626000-08:00"
        }
    }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/mw7dn3qxvkfh8ztc. Vous en aurez besoin à l'étape suivante.

Générer un DCVV

Example

Dans cet exemple, nous allons générer un DCVV pour une transaction EMV sans contact. Les entrées incluent le PAN, le numéro de séquence PAN, le compteur de transactions d'application, le nombre imprévisible et les données de suivi.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
    --primary-account-number=5111112627662122 \
    --generation-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,TrackData
 \
    --validation-data-length 5
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2",
    "ValidationData": "36667"
}
```

Valider DCVV

Example

Dans cet exemple, nous allons valider un DCVV fourni lors d'une transaction. Les mêmes entrées utilisées pour la génération doivent être fournies pour la validation.

Si AWS Payment Cryptography est en mesure de valider, un http/200 est renvoyé. Si la valeur n'est pas validée, elle renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
    --primary-account-number=5111112627662122 \
    --validation-data=36667 \
    --verification-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,TrackData
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2"
}
```

Fonctions spécifiques à Mastercard

Rubriques

- DCVC3
- ARQC -/ CVN14CVN15
- ARQC -/ CVN12CVN13
- 3DS AAV SPA2

DCVC3

DCVC3 est antérieur aux CVN12 systèmes EMV CSK et Mastercard et représente une autre approche d'utilisation des clés dynamiques. Il est parfois également réutilisé pour d'autres cas d'utilisation. Dans ce schéma, les entrées sont les données PAN, PSN, Track1/Track2, un nombre imprévisible et un compteur de transactions (ATC).

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"DCVC3"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
hrh6qgbi3sk4y3wq",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": false,
                        "Sign": false,
                        "Verify": false,
                        "DeriveKey": true,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "08D7B4",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east- 2:111122223333:key/hrh6qgbi3sk4y3wq. Vous en aurez besoin à l'étape suivante.

Générez un DCVC3

Example

Bien qu'il DCVC3 soit généralement généré par une carte à puce, il peut également être généré manuellement, comme dans cet exemple

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk
 --primary-account-number=5413123456784808 --generation-attributes
 DynamicCardVerificationCode='{ApplicationTransactionCounter=0000,TrackData=524106000000069D13
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
    "KeyCheckValue": "08D7B4",
    "ValidationData": "865"
  }
```

Validez le DCVC3

Example

Dans cet exemple, nous allons valider un DCVC3. Notez que l'ATC doit être fourni sous forme de numéro hexadécimal, par exemple un compteur de 11 doit être représenté par 000B. Le service attend une valeur à 3 chiffres DCVC3, donc si vous avez enregistré une valeur à 4 (ou 5) chiffres, tronquez simplement les caractères de gauche jusqu'à obtenir 3 chiffres (par exemple, 15321 devrait entraîner une valeur de donnée de validation de 321).

Si AWS Payment Cryptography est en mesure de valider, un http/200 est renvoyé. Si la valeur n'est pas validée, elle renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk
 --primary-account-number=5413123456784808 --verification-attributes
 DynamicCardVerificationCode='{ApplicationTransactionCounter=000B,TrackData=524106000000069D13
 --validation-data 398
```

```
{
```

```
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
    "KeyCheckValue": "08D7B4"
  }
```

ARQC -/ CVN14CVN15

CVN14 et CVN15 utilisez la méthode EMV CSK de dérivation de clés. Les données de transaction exactes varient entre ces deux méthodes. Consultez la documentation du schéma pour plus de détails sur la création du champ de données de transaction.

ARQC -/ CVN12CVN13

CVN12 et CVN13 sont une ancienne méthode spécifique à MasterCard pour les transactions EMV qui incorpore un nombre imprévisible dans la dérivation par transaction et utilise également une charge utile différente. Pour plus d'informations sur le contenu de la charge utile, veuillez contacter le programme.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN12"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": false,
                        "Sign": false,
```

```
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "08D7B4",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
            "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
        }
    }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/pw3s6nl62t5ushfk. Vous en aurez besoin à l'étape suivante.

Valider l'ARQC

Example

Dans cet exemple, nous allons valider un ARQC généré à l'aide de Mastercard CVN12.

Si AWS Payment Cryptography est en mesure de valider l'ARQC, un http/200 est renvoyé. Si l'arqc n'est pas validé, il renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram 31BE5D49F14A5F01 \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
  --major-key-derivation-mode EMV_OPTION_A \
  --transaction-data
 00000000170000000000000000840008000800008401605170000000009380000B1F220103000000000000000000000
 \
  --session-key-derivation-attributes='{"MastercardSessionKey":
{"ApplicationTransactionCounter":"000B","PanSequenceNumber":"01","PrimaryAccountNumber":"541312
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
```

```
    "KeyCheckValue": "08D7B4"
  }
```

3DS AAV SPA2

SPA2(Application de paiement sécurisé) L'AAV (Account Authentication Value) est utilisée pour les transactions Mastercard 3DS (également connue sous le nom de Mastercard Idenity Check). Il fournit une authentification cryptographique pour les transactions de commerce électronique à l'aide de la génération MAC basée sur HMAC. L'AAV est généré à l'aide de données spécifiques à la transaction et d'une clé secrète partagée.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=HMAC_SHA256,KeyUsage=TR31_M7_HMAC_KEY,KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Gene
 --tags='[{"Key":"KEY_PURPOSE","Value":"SPA2_AAV"},
{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
q5vjtshsg67cz5gn",
                "KeyAttributes": {
                    "KeyUsage": "TR31_M7_HMAC_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "HMAC_SHA256",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
```

```
                "KeyCheckValue": "C661F9",
                "KeyCheckValueAlgorithm": "HMAC",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-west - 2:111122223333:key/q5vjtshsg67cz5gn. Vous en aurez besoin à l'étape suivante.

Générer SPA2 AAV

Example

Dans cet exemple, nous allons générer le composant Issuer Authentication Value (IAV) de l' SPA2 AAV à l'aide de la génération HMAC MAC. Les données du message contiennent les informations spécifiques à la transaction qui seront authentifiées. Le format des données du message doit suivre les SPA2 spécifications de Mastercard et n'est pas couvert dans cet exemple.

> **Note**
>
> Veuillez consulter les spécifications de votre carte Mastercard pour connaître le formatage permettant d'insérer l'IAV dans la valeur AAV.

```
$ aws payment-cryptography-data generate-mac --key-identifier arn:aws:payment-
cryptography:us-west-2:111122223333:key/q5vjtshsg67cz5gn --message-data
 "2226400099919520FFFFd8b448be65694fe7b42f836bad396e9d" --generation-attributes
 Algorithm=HMAC --region us-west-2
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
q5vjtshsg67cz5gn",
    "KeyCheckValue": "C661F9",
    "Mac": "6FB2405E9D8A4C1F7B173F73ADD1A6DC358531CAB0E9994FC5B62012ADDE91FC"
}
```

Vérifiez SPA2 AAV

Example

Dans cet exemple, nous allons vérifier un SPA2 AAV. Les mêmes données de message et la même valeur MAC sont fournies à des fins de vérification.

Si AWS Payment Cryptography est en mesure de valider le MAC, un http/200 est renvoyé. Si le MAC n'est pas validé, il renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-mac --key-identifier arn:aws:payment-
cryptography:us-west-2:111122223333:key/q5vjtshsg67cz5gn --message-
data "2226400099919520FFFFd8b448be65694fe7b42f836bad396e9d" --mac
 "6FB2405E9D8A4C1F7B173F73ADD1A6DC358531CAB0E9994FC5B62012ADDE91FC" --verification-
attributes Algorithm=HMAC --region us-west-2
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
q5vjtshsg67cz5gn",
    "KeyCheckValue": "C661F9"
}
```

Fonctions spécifiques à American Express

Rubriques

- CSC1
- CSC2
- iCSC
- 3DS AEVV

CSC1

La version 1 du CSC est également connue sous le nom d'algorithme CSC classique. Le service peut le fournir sous forme de numéro à 3,4 ou 5 chiffres.

Pour tous les paramètres disponibles, voir AmexCardSecurityCodeVersion1 dans le guide de référence de l'API.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CSC1"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq",
                    "KeyAttributes": {
                        "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
                        "KeyClass": "SYMMETRIC_KEY",
                        "KeyAlgorithm": "TDES_2KEY",
                        "KeyModesOfUse": {
                            "Encrypt": false,
                            "Decrypt": false,
                            "Wrap": false,
                            "Unwrap": false,
                            "Generate": true,
                            "Sign": false,
                            "Verify": true,
                            "DeriveKey": false,
                            "NoRestrictions": false
                        }
                    },
                    "KeyCheckValue": "8B5077",
                    "KeyCheckValueAlgorithm": "ANSI_X9_24",
                    "Enabled": true,
                    "Exportable": true,
                    "KeyState": "CREATE_COMPLETE",
                    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                    "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                    "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
                }
            }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/esh6hn7pxdtttzgq. Vous en aurez besoin à l'étape suivante.

Générez un CSC1

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion1='{CardExpiryDate=1224}' --validation-data-length 4
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq",
    "KeyCheckValue": "8B5077",
    "ValidationData": "3938"
  }
```

Validez le CSC1

Example

Dans cet exemple, nous allons valider un CSC1.

Si AWS Payment Cryptography est en mesure de valider, un http/200 est renvoyé. Si la valeur n'est pas validée, elle renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/esh6hn7pxdtttzgq
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion1='{CardExpiryDate=1224}''  --validation-data 3938
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq",
    "KeyCheckValue": "8B5077"
  }
```

CSC2

La version 2 du CSC est également connue sous le nom d'algorithme CSC amélioré. Le service peut le fournir sous forme de numéro à 3,4 ou 5 chiffres. Le code de service pour CSC2 est généralement 000.

Pour tous les paramètres disponibles, reportez-vous à la section AmexCardSecurityCodeVersion2 du guide de référence de l'API.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CSC2"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
erlm445qvunmvoda",
                "KeyAttributes": {
                    "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "BF1077",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
            }
        }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/erlm445qvunmvoda. Vous en aurez besoin à l'étape suivante.

Générez un CSC2

Dans cet exemple, nous allons générer un CSC2 avec une longueur de 4. Le CSC peut être généré avec une longueur de 3,4 ou 5. Pour American Express, PANs il doit comporter 15 chiffres et commencer par 34 ou 37. La date d'expiration est généralement au format YYMM. Le code de service peut varier. Consultez votre manuel, mais les valeurs typiques sont 000, 201 ou 702

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
erlm445qvunmvoda --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=2412,ServiceCode=000}' --validation-
data-length 4
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda",
"KeyCheckValue": "BF1077",
"ValidationData": "3982"
}
```

Validez le CSC2

Example

Dans cet exemple, nous allons valider un CSC2.

Si AWS Payment Cryptography est en mesure de valider, un http/200 est renvoyé. Si la valeur n'est pas validée, elle renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=2412,ServiceCode=000}' --validation-data
 3982
```

```
{
```

```
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda",
"KeyCheckValue": "BF1077"
}
```

iCSC

L'iCSC est également connu sous le nom d'algorithme CSC statique et est calculé à l'aide de CSC version 2. Le service peut le fournir sous forme de numéro à 3,4 ou 5 chiffres.

Utilisez le code de service 999 pour calculer l'iCSC pour une carte de visite. Utilisez le code de service 702 pour calculer l'iCSC pour une carte sans contact.

Pour tous les paramètres disponibles, reportez-vous à la section AmexCardSecurityCodeVersion2 du guide de référence de l'API.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CSC1"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
        "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/7vrybrbvjcvwtunv",
        "KeyAttributes": {
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
```

```
            },
        },
        "KeyCheckValue": "7121C7",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "CreateTimestamp": "2025-01-29T09:19:21.209000-05:00",
        "UsageStartTimestamp": "2025-01-29T09:19:21.192000-05:00"
        }
    }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east- 1:111122223333:key/7vrybrbvjcvwtunv. Vous en aurez besoin à l'étape suivante.

Générer un iCSC

Dans cet exemple, nous allons générer un iCSC d'une longueur de 4, pour une carte sans contact utilisant le code de service 702. Le CSC peut être généré avec une longueur de 3,4 ou 5. Pour American Express, PANs il doit comporter 15 chiffres et commencer par 34 ou 37.

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv
 --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1224,ServiceCode=702}' --validation-
data-length 4
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv,
    "KeyCheckValue": 7121C7,
    "ValidationData": "2365"
}
```

Valider l'iCSC

Example

Dans cet exemple, nous allons valider un iCSC.

Si AWS Payment Cryptography est en mesure de valider, un http/200 est renvoyé. Si la valeur n'est pas validée, elle renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1224,ServiceCode=702}' --validation-data
 2365
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv,
    "KeyCheckValue": 7121C7
}
```

3DS AEVV

3DS AEVV (3-D Secure Account Verification Value) est utilisé pour l'authentification 3-D Secure d'American Express. Il utilise le même algorithme, CSC2 mais avec des paramètres d'entrée différents. Le champ de date d'expiration doit être rempli avec un nombre imprévisible (aléatoire), et le code de service se compose du code des résultats de l'authentification AEVV (1 chiffre) et du code d'authentification à deuxième facteur (2 chiffres). La longueur de sortie doit être de 3 chiffres.

Pour tous les paramètres disponibles, reportez-vous à la section AmexCardSecurityCodeVersion2 du guide de référence de l'API.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"3DS_AEVV"},
{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
        "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kw8djn5qxvfh3ztm",
        "KeyAttributes": {
```

```
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
        },
        "KeyCheckValue": "8F3A21",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "CreateTimestamp": "2025-02-02T10:30:15.209000-05:00",
        "UsageStartTimestamp": "2025-02-02T10:30:15.192000-05:00"
        }
    }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/kw8djn5qxvfh3ztm. Vous en aurez besoin à l'étape suivante.

Générer un 3DS AEVV

Dans cet exemple, nous allons générer un 3DS AEVV d'une longueur de 3. Le champ de date d'expiration contient un nombre imprévisible (aléatoire) (par exemple, 1234), et le code de service se compose du code de résultats d'authentification AEVV (1 chiffre) et du code d'authentification à deuxième facteur (2 chiffres), par exemple 543 où 5 est le code des résultats d'authentification et 43 est le code d'authentification à deuxième facteur. Pour American Express, PANs il doit comporter 15 chiffres et commencer par 34 ou 37.

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
```

```
kw8djn5qxvfh3ztm --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1234,ServiceCode=543}' --validation-
data-length 3
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm,
    "KeyCheckValue": 8F3A21,
    "ValidationData": "921"
}
```

Validez le 3DS AEVV

Example

Dans cet exemple, nous allons valider un 3DS AEVV.

Si AWS Payment Cryptography est en mesure de valider, un http/200 est renvoyé. Si la valeur n'est pas validée, elle renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1234,ServiceCode=543}' --validation-data
 921
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm,
    "KeyCheckValue": 8F3A21
}
```

Fonctions spécifiques au JCB

Rubriques

- ARQC - CVN04
- ARQC - CVN01

ARQC - CVN04

JCB CVN04 utilise la méthode CSK de dérivation des clés. Consultez la documentation du schéma pour plus de détails sur la création du champ de données de transaction.

ARQC - CVN01

CVN01 est une ancienne méthode JCB pour les transactions EMV qui utilise la dérivation de clé par carte plutôt que la dérivation de session (par transaction) et utilise également une charge utile différente. Ce message est également utilisé par Visa, c'est pourquoi le nom de l'élément porte ce nom même s'il est également utilisé pour JCB. Pour plus d'informations sur le contenu de la charge utile, veuillez contacter la documentation du schéma.

Créer une clé

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN10"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

La réponse renvoie les paramètres de la demande, y compris un ARN pour les appels suivants ainsi qu'une valeur de contrôle clé (KCV).

```
{
                        "Key": {
                            "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/pw3s6nl62t5ushfk",
                            "KeyAttributes": {
                                "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                                "KeyClass": "SYMMETRIC_KEY",
                                "KeyAlgorithm": "TDES_2KEY",
                                "KeyModesOfUse": {
                                    "Encrypt": false,
                                    "Decrypt": false,
                                    "Wrap": false,
                                    "Unwrap": false,
                                    "Generate": false,
                                    "Sign": false,
                                    "Verify": false,
                                    "DeriveKey": true,
                                    "NoRestrictions": false
                                }
                            },
                            "KeyCheckValue": "08D7B4",
                            "KeyCheckValueAlgorithm": "ANSI_X9_24",
                            "Enabled": true,
                            "Exportable": true,
                            "KeyState": "CREATE_COMPLETE",
```

```
                        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                        "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                        "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
                    }
                }
```

Prenez note de `KeyArn` ce qui représente la clé, par exemple arn:aws:payment-cryptography:us-east - 2:111122223333:key/pw3s6nl62t5ushfk. Vous en aurez besoin à l'étape suivante.

Valider l'ARQC

Example

Dans cet exemple, nous allons valider un ARQC généré à l'aide de CVN01 JCB. Cela utilise les mêmes options que la méthode Visa, d'où le nom du paramètre.

Si AWS Payment Cryptography est en mesure de valider l'ARQC, un http/200 est renvoyé. Si l'arqc n'est pas validé, il renverra une réponse http/400.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram D791093C8A921769 \
             --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
             --major-key-derivation-mode EMV_OPTION_A \
             --transaction-data
 00000000170000000000000008400080008000084016051700000000093800000B03011203000000 \
             --session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
             ,"PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                "KeyCheckValue": "08D7B4"
             }
```

Facilitateurs d'acquisition et de paiement

Les acquéreurs PSPs et les facilitateurs de paiement ont généralement des exigences cryptographiques différentes de celles des émetteurs. Cas d'utilisation courants :

Déchiffrement des données

Les données (en particulier les données panoramiques) peuvent être cryptées par un terminal de paiement et doivent être déchiffrées par le backend. Le déchiffrement des données et le chiffrement des données prennent en charge diverses méthodes, notamment les techniques de dérivation TDES, AES et DUKPT. Le service AWS de cryptographie des paiements lui-même est également conforme à la norme PCI P2PE et est enregistré en tant que composant de déchiffrement PCI P2PE.

TranslatePin

Pour garantir la conformité au code PIN PCI, les systèmes d'acquisition ne doivent pas afficher le code PIN du titulaire de la carte en clair une fois qu'il a été saisi sur un appareil sécurisé. Par conséquent, pour transmettre le code PIN du terminal à un système en aval (tel qu'un réseau de paiement ou un émetteur), il est nécessaire de le rechiffrer à l'aide d'une clé différente de celle utilisée par le terminal de paiement. Translate Pin y parvient en convertissant un code PIN crypté d'une clé à une autre en toute sécurité avec le servicebbb. À l'aide de cette commande, les broches peuvent être converties entre différents schémas tels que la dérivation TDES, AES et DUKPT et les formats de blocs de broches tels que ISO-0, ISO-3 et ISO-4.

VerifyMac

Les données d'un terminal de paiement peuvent être enregistrées sur MAC pour s'assurer qu'elles n'ont pas été modifiées pendant le transport. Verify Mac et GenerateMac prend en charge diverses techniques utilisant des clés symétriques, notamment les techniques de dérivation TDES, AES et DUKPT à utiliser avec l'algorithme 1 ISO-9797-1, l'algorithme ISO-9797-1 3 (Retail MAC) et les techniques CMAC.

Rubriques supplémentaires

- Utilisation de touches dynamiques

Utilisation de touches dynamiques

Les clés dynamiques permettent d'utiliser des clés à usage unique ou limité pour des opérations cryptographiques telles que`EncryptData`. Ce flux peut être utilisé lorsque le matériel clé change fréquemment (par exemple à chaque transaction par carte) et que l'on souhaite éviter d'importer le matériel clé dans le service. Les clés de courte durée peuvent être utilisées dans le cadre de SoftPOS/MPOC ou d'autres solutions.

> **Note**
>
> Cela peut être utilisé à la place du flux classique utilisant la cryptographie des AWS paiements, dans lequel les clés cryptographiques sont créées ou importées dans le service et les clés sont spécifiées à l'aide d'un alias de clé ou d'un arn de clé.

Les opérations suivantes prennent en charge les clés dynamiques :

- EncryptData
- DecryptData
- ReEncryptData
- TranslatePin

Déchiffrement de données

L'exemple suivant montre l'utilisation de clés dynamiques avec la commande de déchiffrement. Dans ce cas, l'identifiant de clé est la clé d'encapsulation (KEK) qui sécurise la clé de déchiffrement (fournie dans le paramètre wrapped-key au format TR-31). La clé encapsulée doit être l'objectif principal de D0 à utiliser avec la commande de déchiffrement ainsi qu'un mode d'utilisation de B ou D.

Example

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza
 --cipher-text 1234123412341234123412341234123A --decryption-attributes
 'Symmetric={Mode=CBC,InitializationVector=1234123412341234}'   --wrapped-key
 WrappedKeyMaterial={"Tr31KeyBlock"="D0112D0TN00E0000B05A6E82D7FC68B95C84306634B0000DA4701BE9BC
```

```
{
   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
   "KeyCheckValue": "0A3674",
   "PlainText": "2E138A746A0032023BEF5B85BA5060BA"
}
```

Traduire une épingle

L'exemple suivant montre l'utilisation de touches dynamiques associées à la commande translate pin pour passer d'une clé dynamique à une clé de travail semi-statique pour acquéreur (AWK). Dans ce cas, l'identifiant de clé entrante est la clé d'encapsulation (KEK) qui protège la clé de chiffrement dynamique par code PIN (PEK) fournie au format TR-31. La clé encapsulée doit avoir un objectif clé `P0` ainsi qu'un mode d'utilisation de B ou D. L'identifiant de clé sortante est une clé de type `TR31_P0_PIN_ENCRYPTION_KEY` et un mode d'utilisation de Encrypt=True, Wrap=True

Example

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-block
 "C7005A4C0FA23E02" --incoming-translation-
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}'
 --incoming-key-identifier alias/PARTNER1_KEK  --outgoing-key-
identifier alias/ACQUIRER_AWK_PEK --outgoing-translation-attributes
 IsoFormat0="{PrimaryAccountNumber=171234567890123}"  --incoming-wrapped-key
 WrappedKeyMaterial={"Tr31KeyBlock"="D0112P0TB00S0000EB5D8E63076313162B04245C8CE351C956EA4A16CC
```

```
{
   "PinBlock": "2E66192BDA390C6F",
   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
   "KeyCheckValue": "0A3674"
}
```

Fonctionnalités spécifiques à la région pour la cryptographie des AWS paiements

Certaines fonctionnalités peuvent être spécifiques à une région et ne pas être utilisées d'une autre manière. Ces fonctionnalités sont décrites plus en détail dans cette section.

AS2805

La norme australienne 2805 AS28 (05) est une norme pour les transferts électroniques de fonds utilisée principalement pour les transactions de paiement par carte. Il est géré par Standards Australia. La norme comprend 6 livres qui couvrent de nombreux sujets allant du format des messages aux normes de cryptage.

La partie 6 fournit des conseils sur la gestion des clés, y compris host-to-host (node-to-node) la communication et les exigences cryptographiques pertinentes, tandis que d'autres aspects sont abordés dans d'autres parties. Toute la cryptographie de cette norme est actuellement basée sur le TDES.

> **Note**
>
> AS2805 est actuellement disponible dans la région ap-southeast-2. Il sera déployé dans d'autres régions dans un futur proche.

AS2805 présente un certain nombre de différences par rapport aux autres implémentations, qui sont résumées ci-dessous.

Protection des clés

S'appuie sur des variantes de touches plutôt que sur des blocs-touches comme dans le TR-31/X9.143. AWS La cryptographie des paiements stocke toutes les clés sous forme de blocs clés en interne, mais permet l'importation, l'exportation et le calcul en utilisant AS28 05 variantes définies.

Clés unidirectionnelles

AS2805 impose l'utilisation de touches unidirectionnelles. Si les deux nœuds doivent générer des codes d'authentification des messages (MAC), ils utilisent deux clés.

Pins à épingles

AS2805 définit une technique de dérivation de clés pour des clés de chiffrement par code PIN uniques par transaction. Cela peut être utilisé à la place du DUKPT. Le schéma AS28 05 repose sur les données de transaction (numéro de trace et montant de la transaction) par rapport à l'utilisation du compteur de transactions par le DUKPT.

Validation des échanges de clés

Définit un processus pour valider le KEK avant de commencer à échanger des clés de travail telles que des clés PIN. Dans d'autres systèmes, les KEK sont rarement échangés et sont validés à l'aide du KCV.

AS2805 utilise le concept de variantes de clés plutôt que de blocs de clés pour garantir que les clés ne sont utilisées que dans le but prévu (et unique). Voici comment la cryptographie des AWS paiements établit une correspondance entre les variantes et les blocs de touches lors de l'importation, de l'exportation ou de l'exécution d'autres fonctions cryptographiques à l'aide de clés.

AS2805 Type de clé	AWS Type de clé de cryptographie de paiement
TERMINAL_MAJOR_KEY_VARIANT_00	TR31_K0_KEY_CRYPTION_KEY
VARIANTE DE LA CLÉ DE CHIFFREMENT PAR CODE PIN 28	TR31CLÉ DE CHIFFREMENT _P0_PIN
VARIANTE_CLÉ_D' AUTHENTIFICATION DU MESSAGE_24	TR31_M0_ISO_16609_MAC_KEY
VARIANTE_CLÉ_CHIFFRE_DONNÉES_22	TR31_D0_CLÉ_DE CHIFFREMENT DES DONNÉES SYMÉTRIQUES
VARIANT_MASK_82, VARIANT_MASK_82C0	Options disponibles dans le cadre du processus de validation KEK. Ces types de clés sont éphémères et ne sont pas stockés par le service.

Étant donné deux nœuds, node1 et node2, les exemples suivants sont du point de vue du nœud 1. AWS La cryptographie des paiements prend APIs en charge les deux côtés du processus.

Rubriques

Échange de clé initiale (KEK)

En AS28 2005, chaque camp possède son propre KEK. KEK (s) fait référence à la clé côté expéditeur qui sera utilisée chaque fois que le côté expéditeur aura besoin de protect/wrap clés et les enverra au nœud 2. KEK (r) est la clé créée par le côté opposé (node2).

> **Note**
>
> Ces termes sont relatifs : un côté crée une clé (côté émetteur) et l'autre la reçoit. Ainsi donné KEY1, il est appelé KEK (s) sur le nœud 1 et KEK (r) sur le nœud 2.

Les KEK pour AS28 05 sont toujours de type clé = TR31 _K0_KEY_ENCRYPTION_KEY car ils sont utilisés pour protéger les cryptogrammes et non les blocs de clés. Cela correspond à TERMINAL_MAJOR_KEY_VARIANT_00 tel que défini dans 05 6.1 AS28

Étapes :

1. Création d'une clé

   Créez une clé à l'aide de l'CreateKeyAPI. Vous allez créer une clé de type TR31 _K0_KEY_ENCRYPTION_KEY

2. Déterminer la méthode d'échange de clés avec le nœud 2

   Déterminez comment échanger des KEK avec la contrepartie. Pour AS28 2005, la méthode la plus courante et la plus interopérable est RSA Wrap.

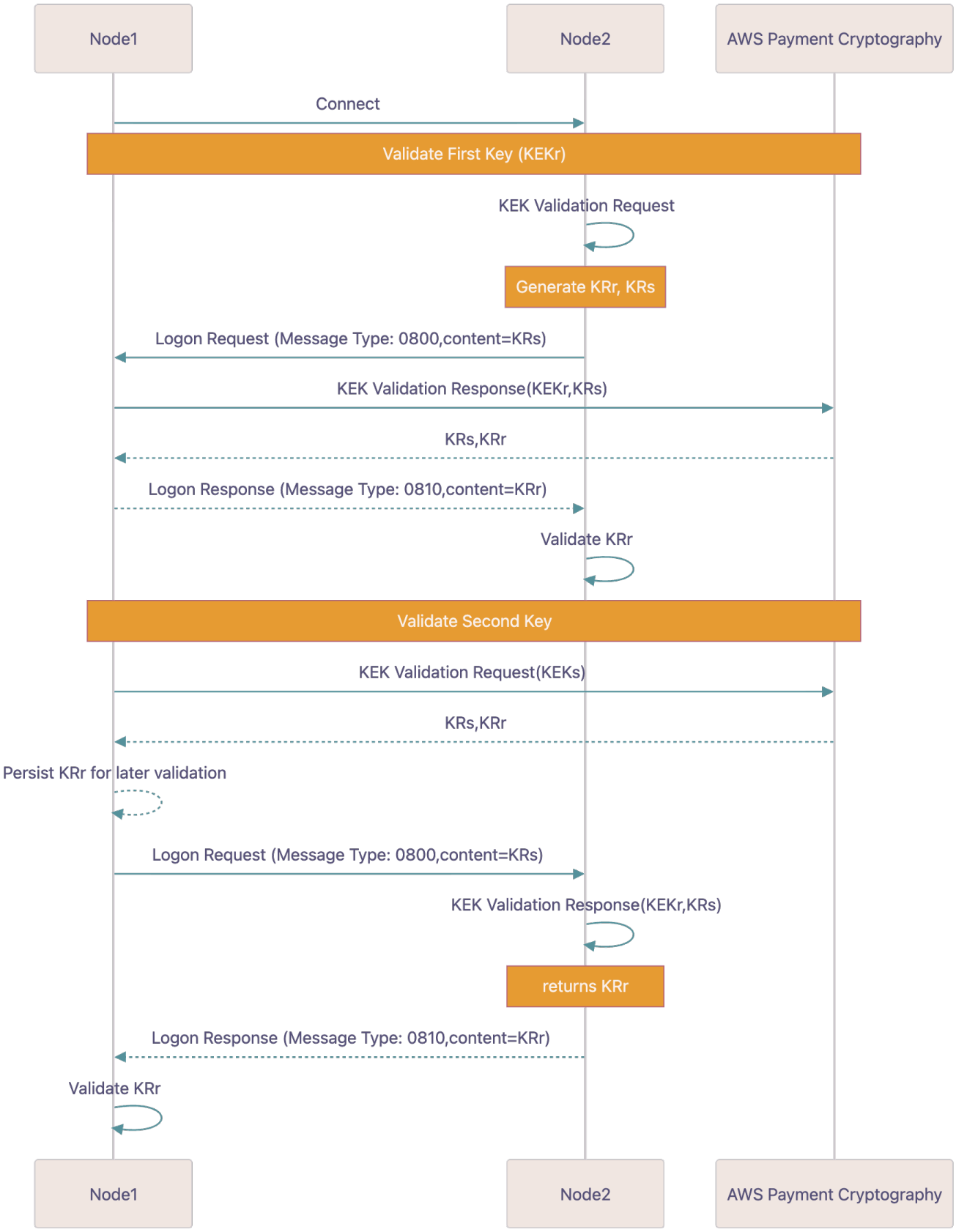
3. Exportation KEKs

   Sur la base de votre sélection ci-dessus, vous recevrez un certificat de clé publique de la part de node2. Vous exécuterez l'exportation à l'aide de ce certificat pour protéger la clé (ou vous en déduirez une clé si vous utilisez l'ECDH).

4. Importer KEKr

   Sur la base de votre sélection ci-dessus, vous allez envoyer un certificat de clé publique à node2. Vous exécuterez l'importation en utilisant ce certificat pour charger les nœuds 2 KEKr dans le service.

Validation du KEK



Lorsque votre service (node1) se connecte au nœud 2, chaque partie s'assure qu'elle utilise le même KEK pour les opérations suivantes à l'aide d'un processus appelé validation KEK.

1. Étapes pour valider la première clé

1.1 Recevoir KRs

Node2 générera un KRs et vous l'enverra dans le cadre du processus de connexion. Ils peuvent utiliser la cryptographie des AWS paiements pour générer cette valeur ou une autre solution.

1.2 Générer une réponse de validation KEK

Votre nœud générera une réponse de validation KEK avec des entrées telles que le KEK (r) et celui KRs fourni à l'étape 1.

Example

```
cat >> generate-kek-validation-response.json
{
    "KekValidationType": {
        "KekValidationResponse": {
            "RandomKeySend": "9217DC67B8763BABCFDF3DADFCD0F84A"
        }
    },
    "RandomKeySendVariantMask": "VARIANT_MASK_82",
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza"
}
```

```
$ aws payment-cryptography-data generate-as2805-kek-validation --cli-input-json
 file://generate-kek-validation-response.json
```

```
{
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
 "KeyCheckValue": "0A3674",
 "RandomKeyReceive": "A4B7E249C40C98178C1B856DB7FB76EB",
 "RandomKeySend": "9217DC67B8763BABCFDF3DADFCD0F84A"
}
```

1.3 Rendement calculé KRr

Renvoie le résultat calculé KRr à node2. Ce nœud la comparera à la valeur calculée à l'étape 1.

2. Étapes pour valider la deuxième clé

2.1 Générer KRr et KRs

Votre nœud générera une valeur aléatoire et une copie inversée (inversée) de cette valeur à l'aide de la cryptographie des AWS paiements. Le service produira ces deux valeurs encapsulées par le ou les KEK. Ils sont appelés KR (s) et KR (r).

Example

```
cat >> generate-kek-validation-request.json
{
    "KekValidationType": {
        "KekValidationRequest": {
            "DeriveKeyAlgorithm": "TDES_2KEY"
        }
    },
"RandomKeySendVariantMask": "VARIANT_MASK_82",
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv"
}
```

```
$ aws payment-cryptography-data generate-as2805-kek-validation --cli-input-json
 file://generate-kek-validation-request.json
```

```
{
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv",
 "KeyCheckValue": "DC1081",
 "RandomKeyReceive": "A4B7E249C40C98178C1B856DB7FB76EB",
 "RandomKeySend": "9217DC67B8763BABCFDF3DADFCD0F84A"
}
```

2.2 Envoyer KRs vers node2

Envoyez-le KRs au nœud 2. Conservez-le KRr pour une validation ultérieure.

2.3 Node2 génère une réponse de validation KEK

Node2 utilise le KEKr et KRs, le génère KRr et le renvoie à votre service.

2.4 Valider la réponse

Comparez KRr la valeur obtenue à l'étape 1 et la valeur renvoyée à l'étape 3. S'ils correspondent, continuez.

Création et transmission de clés de travail

Les clés de travail typiques utilisées en AS28 05 incluent deux jeux de clés :

Clés entre les nœuds, telles que : clé PIN de zone (ZPK), clé de chiffrement de zone (ZEK) et clé d'authentification de zone (ZAK).

Clés entre les terminaux et les nœuds, telles que : touche principale du terminal (TMK) et touche PIN du terminal (TPK) si vous n'utilisez pas DUKPT.

> **Note**
>
> Nous recommandons de minimiser le nombre de clés par terminal et de tirer parti de techniques telles que le TR-34 et le DUKPT dans la mesure du possible qui utilisent un plus petit nombre de clés.

Example

Dans cet exemple, nous avons utilisé des balises facultatives pour suivre l'objectif et l'utilisation de cette clé. Les balises ne sont pas utilisées dans le cadre de la fonction cryptographique du système, mais peuvent être utilisées pour la catégorisation, le suivi financier et peuvent être utilisées pour appliquer des politiques IAM.

```
cat >> create-zone-pin-key.json
{
    "KeyAttributes": {
        "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY",
        "KeyClass": "SYMMETRIC_KEY",
        "KeyAlgorithm": "TDES_2KEY",
        "KeyModesOfUse": {
            "Encrypt": true,
            "Decrypt": true,
            "Wrap": true,
            "Unwrap": true,
            "Generate": false,
```

```
            "Sign": false,
            "Verify": false,
            "DeriveKey": false,
            "NoRestrictions": false
        }
    },
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "Exportable": true,
    "Enabled": true,
    "Tags": [
        {
            "Key": "AS2805_KEYTYPE",
            "Value": "ZONE_PIN_KEY_VARIANT28"
        }
    ]
}
```

```
$ aws payment-cryptography-data create-key --cli-input-json file://create-zone-pin-key.json --region ap-southeast-2
```

```
{
 "Key": {
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/alsuwfxug3pgy6xh",
 "KeyAttributes": {
 "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY",
 "KeyClass": "SYMMETRIC_KEY",
 "KeyAlgorithm": "TDES_2KEY",
 "KeyModesOfUse": {
 "Encrypt": true,
 "Decrypt": true,
 "Wrap": true,
 "Unwrap": true,
 "Generate": false,
 "Sign": false,
 "Verify": false,
 "DeriveKey": false,
 "NoRestrictions": false
 }
 },
 "KeyCheckValue": "9A325B",
 "KeyCheckValueAlgorithm": "ANSI_X9_24",
 "Enabled": true,
 "Exportable": true,
```

```
 "KeyState": "CREATE_COMPLETE",
 "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
 "CreateTimestamp": "2025-12-17T09:05:27.586000-08:00",
 "UsageStartTimestamp": "2025-12-17T09:05:27.570000-08:00"
 }
}
```

Exportation de clés de travail

Pour maintenir la compatibilité avec les autres parties, AWS Payment Cryptography prend en charge AS28 05 techniques d'encapsulage de clés symétriques qui utilisent des variantes de clés au lieu de blocs-clés tels que le TR-31. Si plusieurs clés sont partagées entre les parties, chacune doit être exportée individuellement. Si les données sont envoyées de manière bidirectionnelle, il peut y avoir deux clés entre les parties du même type, telles que ZAK (s) et ZAK (r), qui sont utilisées par chaque partie pour générer des codes d'authentification des messages.

Les paramètres supplémentaires à importer et à exporter dans ces formats sont spécifiés dans les commandes.

```
cat >> export-zone-pin-key.json
{
    "ExportKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
    "KeyMaterial": {
        "As2805KeyCryptogram": {
            "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/rhfm6tenpxapkmrv",
            "As2805KeyVariant: "PIN_ENCRYPTION_KEY_VARIANT_28"
        }
    }
}
```

```
$ aws payment-cryptography-data export-key --cli-input-json file://export-zone-pin-
key.json --region ap-southeast-2
```

```
{
    "WrappedKey": {
        "KeyCheckValue": "DC1081",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyMaterial": "HDC10AEF038E695DDD72AF08DC1BB422D",
        "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM",
```

```
        "WrappingKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv"
    }
}
```

Traduction d'épingles

AS2805 décrit un mode de dérivation de clés spécifique à une session dans la section 6.4. Il a un objectif similaire à celui du DUKPT et l'un ou l'autre algorithme peut être utilisé car DUKPT est traité dans la section 6.7. Dans ce schéma, un code PIN de session (connu sous le nom de KPE) est dérivé du code PIN du terminal en utilisant SystemTraceAuditNumber (STAN) et en TransactionAmount tant que données de dérivation.

Translate pin est une fonction courante qui permet de traduire to/from une variété de formats. Dans cet exemple, nous traduisons un code PIN d'un KPE en clé de cryptage par code PIN (PEK), par exemple lors de l'envoi d'un code PIN à un réseau de paiement.

```
cat >> translate-pin-as2805.json
{
    "EncryptedPinBlock": "B3B34B43BAB5F81A",
    "IncomingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "IncomingTranslationAttributes": {
        "IsoFormat0": {
            "PrimaryAccountNumber": "9999179999900013"
        }
    },
      "IncomingAs2805Attributes": {
        "SystemTraceAuditNumber": "000348",
        "TransactionAmount": "000000000328"
    },
    "OutgoingKeyIdentifier": "",
    "OutgoingTranslationAttributes": {
        "IsoFormat0": {
            "PrimaryAccountNumber": "9999179999900013"
        }
    }
}
```

```
$ aws payment-cryptography-data translate-pin-data --cli-input-json file://translate-
pin-as2805.json  --region ap-southeast-2
```

```
{
    "WrappedKey": {
        "KeyCheckValue": "DC1081",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyMaterial": "HDC10AEF038E695DDD72AF08DC1BB422D",
        "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM",
        "WrappingKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv"
    }
}
```

Génération et validation de Mac

Les commandes de génération et de vérification MAC prennent en charge une variété de commandes, MACs notamment HMAC, CMAC, EMV MAC, etc. Pour AS28 2005, il existe une variante supplémentaire définie au AS28 05.4.1. Généralement, en AS28 05, les messages entrants sont vérifiés à l'aide de ce MAC et les messages sortants incluent également un MAC.

```
cat verify-mac.json
{
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "Mac": "86304058",
    "MessageData": "73D8BA54D3852951DAEA41",
    "VerificationAttributes": {
        "Algorithm": "AS2805_4_1"
    }
}
```

```
$ aws payment-cryptography-data verify-mac --cli-input-json file://verify-mac.json --
region ap-southeast-2
```

```
{
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "KeyCheckValue": "2976E7"
}
```

Sécurité dans le domaine de la cryptographie des AWS paiements

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. Third-party les auditeurs 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 la cryptographie des AWS paiements, consultez la section Services AWS concernés par programme de conformité .
- Sécurité dans le cloud : votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette rubrique vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation de la cryptographie des AWS paiements. Il vous explique comment configurer le chiffrement des AWS paiements pour atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources de cryptographie des AWS paiements.

Rubriques

Protection des données dans le cadre de la cryptographie des AWS paiements

Le modèle de responsabilité AWS partagée (modèle de) s'applique à la protection des données dans le domaine de la cryptographie des AWS paiements. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez la FAQ sur la confidentialité des données et les . Pour plus d'informations sur la protection des données en Europe, consultez le Centre du règlement général sur la protection des données (RGPD).

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou Gestion des identités et des accès AWS (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- SSL/TLS À utiliser pour communiquer avec AWS les ressources. Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour capturer AWS des activités, consultez la section Utilisation des CloudTrail sentiers dans le guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3.

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que

le champ Nom. Cela inclut lorsque vous utilisez AWS Payment Cryptography ou une autre méthode à Services AWS l'aide de la console, de l'API ou des AWS SDK. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

AWS Payment Cryptography stocke et protège vos clés de chiffrement des paiements afin de les rendre hautement disponibles tout en vous fournissant un contrôle d'accès solide et flexible.

Rubriques

Protection des éléments de clé

Par défaut, AWS Payment Cryptography protège le contenu des clés cryptographiques pour les clés de paiement gérées par le service. En outre, AWS Payment Cryptography propose des options pour importer des éléments clés créés en dehors du service. Pour obtenir des informations techniques sur les clés de paiement et les éléments clés, consultez les détails cryptographiques d'AWS Payment Cryptography.

Chiffrement des données

Les données d'AWS Payment Cryptography comprennent les clés de chiffrement des paiements AWS, le contenu des clés de chiffrement qu'elles représentent et leurs attributs d'utilisation. Le contenu clé n'existe en texte clair que dans les modules matériels de sécurité (HSM) d'AWS Payment Cryptography et uniquement lorsqu'ils sont utilisés. Dans le cas contraire, le matériel et les attributs clés sont chiffrés et stockés dans un stockage persistant durable.

Le contenu clé généré ou chargé par AWS Payment Cryptography pour les clés de paiement ne quitte jamais les limites des HSM d'AWS Payment Cryptography non chiffrés. Il peut être exporté chiffré par les opérations de l'API AWS Payment Cryptography.

Chiffrement au repos

AWS Payment Cryptography génère des informations clés pour les clés de paiement dans les HSM PCI PTS HSM-listed . 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 de cryptographie de paiement et les clés de chiffrement qui protègent le matériel clé ne quittent jamais les HSM sous forme de texte clair.

Le chiffrement et la gestion des éléments clés pour les clés de chiffrement des paiements sont entièrement gérés par le service.

Pour plus de détails, consultez les détails cryptographiques d'AWS Key Management Service.

Chiffrement en transit

Les éléments clés générés ou chargés AWS par Payment Cryptography pour les clés de paiement ne sont jamais exportés ou transmis dans le cadre des opérations de l'API AWS Payment Cryptography en texte clair. AWS La cryptographie des paiements utilise des identifiants de clé pour représenter les clés dans les opérations d'API.

Cependant, certaines opérations d'API exportent des clés chiffrées par une clé d'échange de clés précédemment partagée ou asymétrique. Les clients peuvent également utiliser les opérations de l'API pour importer des informations clés cryptées pour les clés de paiement.

Tous les appels de l'API de cryptographie des AWS paiements doivent être signés et transmis à l'aide du protocole TLS (Transport Layer Security). AWS La cryptographie des paiements nécessite des versions TLS et des suites de chiffrement définies par la norme PCI comme une « cryptographie forte ». Tous les points de terminaison de service prennent en charge le protocole TLS 1.2—1.3 et le protocole TLS post-quantique hybride.

Pour plus de détails, consultez les détails cryptographiques d'AWS Key Management Service.

Confidentialité du trafic inter-réseau

AWS Payment Cryptography prend en charge une console de gestion AWS et un ensemble d'opérations d'API qui vous permettent de créer et de gérer des clés de paiement et de les utiliser dans des opérations cryptographiques.

AWS La cryptographie des paiements prend en charge deux options de connectivité réseau entre votre réseau privé et AWS.

- Une connexion VPN IPSec sur Internet.
- AWS Direct Connect, qui relie votre réseau interne à un site AWS Direct Connect via un câble Ethernet à fibre optique standard.

Tous les appels de l'API de cryptographie des paiements 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 (HSM) qui stockent les éléments clés pour les clés de paiement est autorisé uniquement à partir d'hôtes d'API AWS Payment Cryptography connus sur le réseau interne d'AWS.

Pour vous connecter directement à AWS Payment Cryptography depuis votre cloud privé virtuel (VPC) sans envoyer de trafic via l'Internet public, utilisez des points de terminaison VPC optimisés par AWS. PrivateLink Pour plus d'informations, consultez Connexion à AWS Payment Cryptography via un point de terminaison VPC.

AWS Payment Cryptography 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 l'API AWS Payment Cryptography.

Résilience dans la cryptographie des AWS paiements

AWS l'infrastructure mondiale est construite autour AWS des régions et des zones de disponibilité. Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par un réseau à latence faible, à débit élevé et à forte redondance. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont davantage disponibles, tolérantes aux pannes et ont une plus grande capacité de mise à l'échelle que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

Isolement régional

AWS Payment Cryptography est un service régional disponible dans plusieurs régions.

La conception isolée par région d'AWS Payment Cryptography garantit qu'un problème de disponibilité dans une région AWS ne peut affecter le fonctionnement d'AWS Payment Cryptography

dans aucune autre région. AWS Payment Cryptography 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 de niveau de service (SLA) d'AWS Payment Cryptography inclut un engagement de service de 99,99 % pour toutes les API de cryptographie des paiements. Pour respecter cet engagement, AWS Payment Cryptography garantit que toutes les données et informations d'autorisation requises pour exécuter une demande d'API sont disponibles sur tous les hôtes régionaux qui reçoivent la demande.

L'infrastructure de chiffrement des paiements AWS est répliquée dans au moins trois zones de disponibilité (AZ) dans chaque région. Pour garantir que les défaillances de plusieurs hôtes n'affectent pas les performances d'AWS Payment Cryptography, AWS Payment Cryptography est conçu pour gérer le trafic client en provenance de n'importe laquelle des AZ d'une région.

Les modifications que vous apportez aux propriétés ou aux autorisations d'une clé de paiement sont reproduites sur tous les hôtes de la région afin de garantir que les demandes ultérieures puissent être traitées correctement par tous les hôtes de la région. Les demandes d'opérations cryptographiques utilisant votre clé de paiement sont transmises à une flotte de modules matériels de sécurité (HSM) AWS Payment Cryptography, chacun d'entre eux pouvant effectuer l'opération avec la clé de paiement.

Multi-tenant design

La conception mutualisée d'AWS Payment Cryptography lui permet de respecter le SLA de disponibilité et de maintenir des taux de demandes élevés, tout en protégeant la confidentialité de vos clés et de vos données.

Plusieurs mécanismes de renforcement de l'intégrité sont déployés pour garantir que la clé de paiement que vous avez spécifiée pour l'opération cryptographique est toujours celle qui est utilisée.

Le contenu clé en texte clair de vos clés de cryptographie de paiement est largement protégé. Le matériel clé est crypté dans le HSM dès sa création, et le matériel clé crypté est immédiatement transféré vers un stockage sécurisé. 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. Les éléments de clé en texte clair ne quittent jamais les HSM ; ils ne sont jamais écrits dans un stockage persistant.

Pour plus d'informations sur les mécanismes utilisés par AWS Payment Cryptography pour sécuriser vos clés, consultez les détails cryptographiques d'AWS Payment Cryptography.

Sécurité de l'infrastructure dans AWS Payment Cryptography

En tant que service géré, AWS Payment Cryptography il est protégé par les procédures de sécurité du réseau AWS mondial décrites dans le livre blanc Amazon Web Services : présentation des processus de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder AWS Payment Cryptography via le réseau. Les clients doivent prendre en charge le protocole TLS (Transport Layer Security) 1.2 ou version ultérieure. Les clients doivent également prendre en charge les suites de chiffrement à parfaite confidentialité (PFS), telles que Ephemeral (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Diffie-Hellman La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Isolement des hôtes physiques

La sécurité de l'infrastructure physique utilisée par AWS Payment Cryptography 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.

Le chiffrement des paiements AWS est pris en charge par des modules de sécurité HSM-listed matériels (HSM) PCI PTS prêts à l'emploi dédiés. Le contenu clé des clés de cryptographie de paiement AWS est stocké uniquement dans la mémoire volatile des HSM, et uniquement lorsque la clé de cryptographie de paiement est utilisée. Les HSM se trouvent dans des racks à accès contrôlé au sein des centres de données Amazon qui appliquent un double contrôle pour tout accès physique. Pour obtenir des informations détaillées sur le fonctionnement des HSM de chiffrement des paiements AWS, consultez la section Détails du chiffrement des paiements AWS.

Connexion à la cryptographie des AWS paiements via un point de terminaison VPC

Vous pouvez vous connecter directement à AWS Payment Cryptography 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 AWS Payment Cryptography s'effectue entièrement au sein du réseau. AWS

AWS Payment Cryptography prend en charge les points de terminaison Amazon Virtual Private Cloud (Amazon VPC) alimentés par. AWS PrivateLink Chaque point de terminaison d'un VPC est représenté par une ou plusieurs interfaces réseau Elastic (ENI) avec des adresses IP privées dans vos sous-réseaux VPC.

Le point de terminaison VPC de l'interface connecte votre VPC directement à AWS Payment Cryptography 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 AWS Payment Cryptography.

Régions

AWS La cryptographie des paiements prend en charge les points de terminaison VPC et les politiques des points de terminaison VPC Régions AWS dans tous les cas où la cryptographie des paiements est prise AWS en charge.

Rubriques

- Considérations relatives aux points de terminaison VPC de cryptographie des AWS paiements
- Création d'un point de terminaison VPC pour AWS la cryptographie des paiements
- Connexion à un point de AWS terminaison VPC de chiffrement des paiements
- Contrôle de l'accès à votre point de terminaison d'un VPC
- Utilisation d'un point de terminaison VPC dans une déclaration de politique
- Journalisation de votre point de terminaison d'un VPC

Considérations relatives aux points de terminaison VPC de cryptographie des AWS paiements

> **Note**
>
> Bien que les points de terminaison VPC vous permettent de vous connecter au service dans une seule zone de disponibilité (AZ), nous vous recommandons de vous connecter à trois zones de disponibilité pour des raisons de haute disponibilité et de redondance.

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

AWS La prise en charge de la cryptographie des paiements pour un point de terminaison VPC inclut les éléments suivants.

- Vous pouvez utiliser votre point de terminaison VPC pour appeler toutes les opérations du plan de contrôle de cryptographie des AWS paiements et toutes les opérations du plan de données AWS de cryptographie des paiements depuis un VPC.
- Vous pouvez créer un point de terminaison VPC d'interface qui se connecte à un point de terminaison de région AWS de cryptographie des paiements.
- AWS La cryptographie des paiements comprend un plan de contrôle et un plan de données. Vous pouvez choisir de configurer un ou les deux sous-services AWS PrivateLink , mais chacun est configuré séparément.
- Vous pouvez utiliser AWS CloudTrail les journaux pour vérifier votre utilisation des clés de chiffrement des AWS paiements via le point de terminaison VPC. Pour en savoir plus, consultez Journalisation de votre point de terminaison d'un VPC.

Création d'un point de terminaison VPC pour AWS la cryptographie des paiements

Vous pouvez créer un point de terminaison VPC pour le chiffrement des AWS paiements à l'aide de la console Amazon VPC ou de l'API Amazon VPC. Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide AWS PrivateLink .

- Pour créer un point de terminaison VPC pour le chiffrement des AWS paiements, utilisez les noms de service suivants :

```
com.amazonaws.region.payment-cryptography.controlplane
```

```
com.amazonaws.region.payment-cryptography.dataplane
```

  Par exemple, dans la région USA Ouest (Oregon) (`us-west-2`), les noms des services seraient les suivants :

```
com.amazonaws.us-west-2.payment-cryptography.controlplane
```

```
com.amazonaws.us-west-2.payment-cryptography.dataplane
```

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 Activer le nom DNS, le nom d'hôte DNS standard AWS de chiffrement des paiements correspond à votre point de terminaison VPC. Par exemple, `https://controlplane.payment-cryptography.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.payment-cryptography.controlplane`.

Cette option facilite l'utilisation du point de terminaison d'un VPC. Les AWS SDK AWS CLI utilisent par défaut le nom d'hôte DNS standard de AWS Payment Cryptography. Vous n'avez donc pas besoin de spécifier l'URL du point de terminaison 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 .

Connexion à un point de AWS terminaison VPC de chiffrement des paiements

Vous pouvez vous connecter à AWS Payment Cryptography via le point de terminaison VPC à l'aide d' AWS un SDK, du ou. AWS CLI 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 payment-cryptography list-keys --endpoint-url https://
vpce-1234abcdf5678c90a-09p7654s-us-east-1a.ec2.us-east-1.vpce.amazonaws.com
```

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 standard AWS de cryptographie des paiements correspond à votre point de terminaison VPC. Les SDK AWS CLI et utilisent ce nom

d'hôte par défaut. Vous pouvez donc commencer à utiliser le point de terminaison VPC pour vous connecter à un point de terminaison régional de cryptographie des AWS paiements 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.

Contrôle de l'accès à votre point de terminaison d'un VPC

Pour contrôler l'accès à votre point de terminaison VPC pour le chiffrement des AWS paiements, associez une politique de point de terminaison VPC à votre point de terminaison VPC. La politique de point de terminaison détermine si les principaux peuvent utiliser le point de terminaison VPC pour AWS appeler des opérations de cryptographie de paiement avec des ressources de cryptographie de paiement AWS spécifiques.

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 .

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.

Rubriques

- À propos des politiques de point de terminaison d'un VPC
- Politique de point de terminaison d'un VPC par défaut
- Création d'une stratégie de point de terminaison de VPC
- Affichage d'une politique de point de terminaison d'un VPC

À propos des politiques de point de terminaison d'un VPC

Pour qu'une demande AWS de cryptographie de paiement utilisant un point de terminaison VPC aboutisse, le principal doit obtenir des autorisations provenant de deux sources :

- Une politique basée sur l'identité doit autoriser le principal à appeler l'opération sur la ressource (clés de chiffrement des AWS paiements 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 en matière de clés peut autoriser un principal à appeler Decrypt pour une clé de cryptographie AWS de paiement particulière. Cependant, la politique du point de terminaison du VPC peut ne pas autoriser ce principal à faire appel à ces clés de cryptographie de AWS paiement en utilisant `Decrypt` le point de terminaison.

Une politique de point de terminaison VPC peut également autoriser un principal à utiliser le point de terminaison pour appeler StopKeyUsagecertaines clés de cryptographie AWS de paiement. Mais si le principal ne dispose pas de ces autorisations dans le cadre d'une politique IAM, la demande échoue.

Politique de point de terminaison d'un VPC par défaut

Chaque point de terminaison d'un VPC dispose d'une politique de point de terminaison d'un VPC, mais vous n'êtes pas tenu de spécifier la politique. Si vous ne spécifiez pas de politique, la politique de point de terminaison par défaut autorise toutes les opérations effectuées par tous les principaux sur toutes les ressources du point de terminaison.

Toutefois, pour les ressources AWS de cryptographie des paiements, le principal doit également être autorisé à appeler l'opération à partir d'une politique IAM. Par conséquent, en pratique, la politique par défaut indique que si un principal a l'autorisation d'appeler une opération sur une ressource, il peut également l'appeler à l'aide du point de terminaison.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Principal": "*",
      "Resource": "*"
    }
  ]
}
```

Pour permettre aux principaux d'utiliser le point de terminaison d'un VPC uniquement pour un sous-ensemble de leurs opérations autorisées, créez ou mettre à jour la politique de point de terminaison d'un VPC.

Création d'une stratégie de point de terminaison de VPC

Une politique de point de terminaison d'un VPC détermine si un principal a l'autorisation d'utiliser le point de terminaison d'un VPC pour effectuer des opérations sur une ressource. Pour les ressources de cryptographie des AWS paiements, le principal doit également être autorisé à effectuer les opérations conformément à une politique IAM.

Chaque instruction de politique de point de terminaison d'un VPC nécessite les éléments suivants :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

L'instruction de politique ne spécifie pas le point de terminaison d'un VPC. Au lieu de cela, elle s'applique à tout point de terminaison d'un VPC auquel la politique est attachée. Pour plus d'informations, consultez Contrôle de l'accès aux services avec points de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC.

Voici un exemple de politique de point de terminaison VPC pour la cryptographie des AWS paiements. Lorsqu'il est connecté à un point de terminaison VPC, cette politique permet d'`ExampleUser`utiliser le point de terminaison VPC pour appeler les opérations spécifiées sur les clés de cryptographie de paiement spécifiées AWS . Avant d'utiliser une politique comme celle-ci, remplacez l'exemple d'identifiant principal et de clé par des valeurs valides provenant de votre compte.

```
{
   "Statement":[
      {
         "Sid": "AllowDecryptAndView",
         "Principal": {"AWS": "arn:aws:iam::111122223333:user/ExampleUser"},
         "Effect":"Allow",
         "Action": [
             "payment-cryptography:Decrypt",
             "payment-cryptography:GetKey",
             "payment-cryptography:ListAliases",
```

```
            "payment-cryptography:ListKeys",
            "payment-cryptography:GetAlias"
         ],
        "Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
      }
   ]
}
```

AWS CloudTrail enregistre toutes les opérations qui utilisent le point de terminaison VPC. Toutefois, vos CloudTrail journaux n'incluent pas les opérations demandées par les principaux sur d'autres comptes ni les opérations relatives aux clés de chiffrement des AWS paiements sur d'autres comptes.

Ainsi, vous souhaiterez peut-être créer une politique de point de terminaison VPC qui empêche les principaux titulaires de comptes externes d'utiliser le point de terminaison VPC pour appeler des opérations de cryptographie de AWS paiement sur n'importe quelle clé du compte local.

L'exemple suivant utilise la clé de condition PrincipalAccount globale aws : pour refuser l'accès à tous les principaux pour toutes les opérations sur toutes les clés de chiffrement des AWS paiements, sauf si le principal se trouve sur le compte local. Avant d'utiliser une politique comme celle-ci, remplacez l'ID de compte d'exemple par un ID valide.

```
{
  "Statement": [
    {
      "Sid": "AccessForASpecificAccount",
      "Principal": {"AWS": "*"},
      "Action": "payment-cryptography:*",
      "Effect": "Deny",
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": "111122223333"
        }
      }
    }
  ]
}
```

Affichage d'une politique de point de terminaison d'un VPC

Pour consulter la politique de point de terminaison VPC d'un point de terminaison, utilisez la console de gestion du VPC ou l'opération. DescribeVpcEndpoints

La AWS CLI commande suivante obtient la politique du point de terminaison avec l'ID de point de terminaison VPC spécifié.

Avant d'utiliser cette commande, remplacez l'exemple d'ID de point de terminaison d'exemple par un ID valide provenant de votre compte.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==`vpce-1234abcdf5678c90a`].[PolicyDocument]'
--output text
```

Utilisation d'un point de terminaison VPC dans une déclaration de politique

Vous pouvez contrôler l'accès aux ressources et aux opérations de cryptographie des AWS paiements lorsque la demande provient d'un VPC ou utilise un point de terminaison VPC. Pour ce faire, utilisez-en une : 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**
>
> La clé de `aws:sourceIP` condition n'est pas effective lorsque la demande provient d'un point de terminaison Amazon VPC. 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 clés de chiffrement des AWS paiements, 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 permet à un utilisateur d'effectuer des opérations cryptographiques particulières avec des clés de cryptographie de AWS paiement uniquement lorsque la demande utilise le point de terminaison VPC spécifié, bloquant ainsi l'accès à la fois depuis Internet et les AWS PrivateLink connexions (si elles sont configurées). Lorsqu'un utilisateur fait une demande à AWS Payment Cryptography, l'ID du point de terminaison VPC figurant dans la demande est comparé à la valeur de `aws:sourceVpce` la clé de condition indiquée 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 les identifiants de point de terminaison VPC par des valeurs valides pour votre compte.

JSON

```
{
    "Id": "example-key-1",
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EnableIAMPolicies",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "payment-cryptography:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RestrictUsageToMyVPCEndpoint",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "payment-cryptography:EncryptData",
                "payment-cryptography:DecryptData"
```

```
            ],
            "Resource": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
*",
            "Condition": {
                "StringNotEquals": {
                    "aws:sourceVpce": "vpce-1234abcdf5678c90a"
                }
            }
        }

    ]
}
```

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

L'exemple de politique de clé suivant autorise les commandes qui gèrent les clés de cryptographie des AWS paiements uniquement lorsqu'elles proviennent`vpc-12345678`. En outre, il autorise les commandes qui utilisent les clés de cryptographie des AWS paiements pour les 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 les identifiants de point de terminaison VPC par des valeurs valides pour votre compte.

JSON

```
{
    "Id": "example-key-2",
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminActionsFromVPC12345678",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
```

```
                "payment-cryptography:Create*",
                "payment-cryptography:Encrypt*",
                "payment-cryptography:ImportKey*",
                "payment-cryptography:GetParametersForImport*",
                "payment-cryptography:TagResource",
                "payment-cryptography:UntagResource"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-12345678"
                }
            }
        },
        {
            "Sid": "AllowKeyUsageFromVPC2b2b2b2b",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
                "payment-cryptography:Encrypt*",
                "payment-cryptography:Decrypt*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-2b2b2b2b"
                }
            }
        },
        {
            "Sid": "AllowListReadActionsFromEverywhere",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
                "payment-cryptography:List*",
                "payment-cryptography:Get*"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Journalisation de votre point de terminaison d'un VPC

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

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 GenerateMac qui utilise le point de terminaison d'un VPC. Le champ `vpcEndpointId` apparaît à la fin de l'entrée de journal.

```
{
      "eventVersion": "1.08",
      "userIdentity": {
          "principalId": "TESTXECZ5U9M4LGF2N6Y5:i-98761b8890c09a34a",
          "arn": "arn:aws:sts::111122223333:assumed-role/samplerole/
i-98761b8890c09a34a",
          "accountId": "111122223333",
          "accessKeyId": "TESTXECZ5U2ZULLHHMJG",
          "sessionContext": {
              "sessionIssuer": {
                  "type": "Role",
                  "principalId": "TESTXECZ5U9M4LGF2N6Y5",
                  "arn": "arn:aws:iam::111122223333:role/samplerole",
                  "accountId": "111122223333",
                  "userName": "samplerole"
              },
              "webIdFederationData": {},
              "attributes": {
                  "creationDate": "2024-05-27T19:34:10Z",
                  "mfaAuthenticated": "false"
              },
              "ec2RoleDelivery": "2.0"
          }
      },
      "eventTime": "2024-05-27T19:49:54Z",
```

```
      "eventSource": "payment-cryptography.amazonaws.com",
      "eventName": "CreateKey",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "172.31.85.253",
      "userAgent": "aws-cli/2.14.5 Python/3.9.16 Linux/6.1.79-99.167.amzn2023.x86_64
 source/x86_64.amzn.2023 prompt/off command/payment-cryptography.create-key",
      "requestParameters": {
          "keyAttributes": {
              "keyUsage": "TR31_M1_ISO_9797_1_MAC_KEY",
              "keyClass": "SYMMETRIC_KEY",
              "keyAlgorithm": "TDES_2KEY",
              "keyModesOfUse": {
                  "encrypt": false,
                  "decrypt": false,
                  "wrap": false,
                  "unwrap": false,
                  "generate": true,
                  "sign": false,
                  "verify": true,
                  "deriveKey": false,
                  "noRestrictions": false
              }
          },
          "exportable": true
      },
      "responseElements": {
          "key": {
              "keyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
              "keyAttributes": {
                  "keyUsage": "TR31_M1_ISO_9797_1_MAC_KEY",
                  "keyClass": "SYMMETRIC_KEY",
                  "keyAlgorithm": "TDES_2KEY",
                  "keyModesOfUse": {
                      "encrypt": false,
                      "decrypt": false,
                      "wrap": false,
                      "unwrap": false,
                      "generate": true,
                      "sign": false,
                      "verify": true,
                      "deriveKey": false,
                      "noRestrictions": false
                  }
```

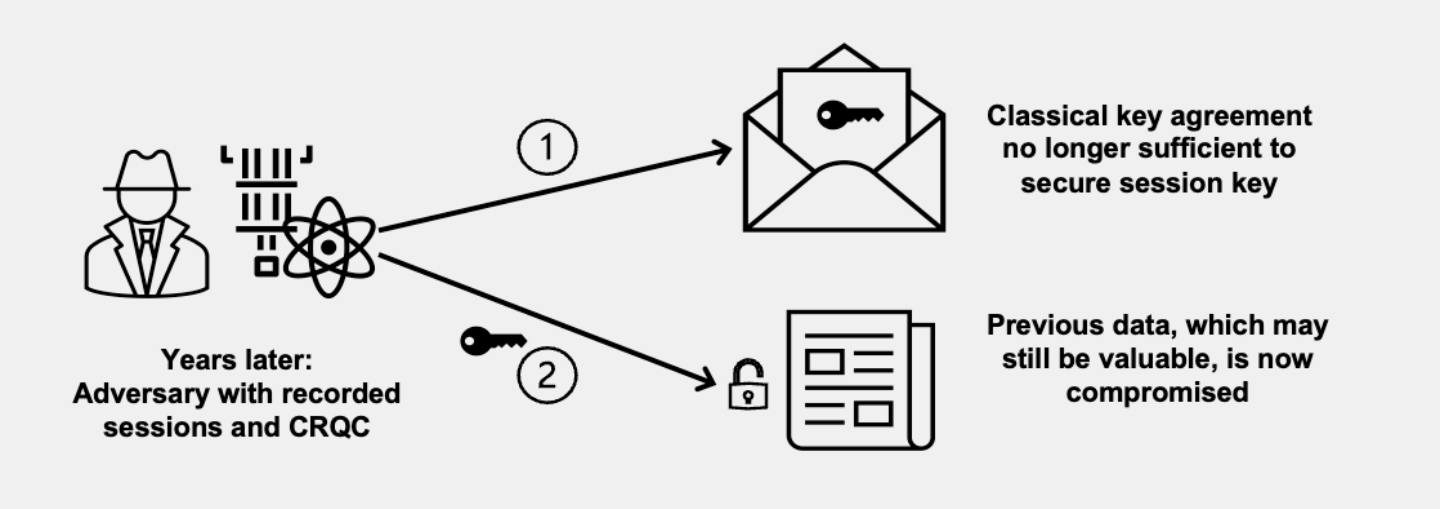
```
            },
            "keyCheckValue": "A486ED",
            "keyCheckValueAlgorithm": "ANSI_X9_24",
            "enabled": true,
            "exportable": true,
            "keyState": "CREATE_COMPLETE",
            "keyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "createTimestamp": "May 27, 2024, 7:49:54 PM",
            "usageStartTimestamp": "May 27, 2024, 7:49:54 PM"
        }
    },
    "requestID": "f3020b3c-4e86-47f5-808f-14c7a4a99161",
    "eventID": "b87c3d30-f3ab-4131-87e8-bc54cfef9d29",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "vpcEndpointId": "vpce-1234abcdf5678c90a",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "vpce-1234abcdf5678c90a-
oo28vrvr.controlplane.payment-cryptography.us-east-1.vpce.amazonaws.com"
    }
}
```


Utilisation du TLS post-quantique hybride

AWS La cryptographie des paiements et de nombreux autres services prennent en charge une option hybride d'échange de clés post-quantique pour le protocole de cryptage réseau TLS (Transport Layer Security). Vous pouvez utiliser cette option TLS lorsque vous vous connectez à des points de terminaison d'API ou lorsque vous utilisez les kits SDK AWS. 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.

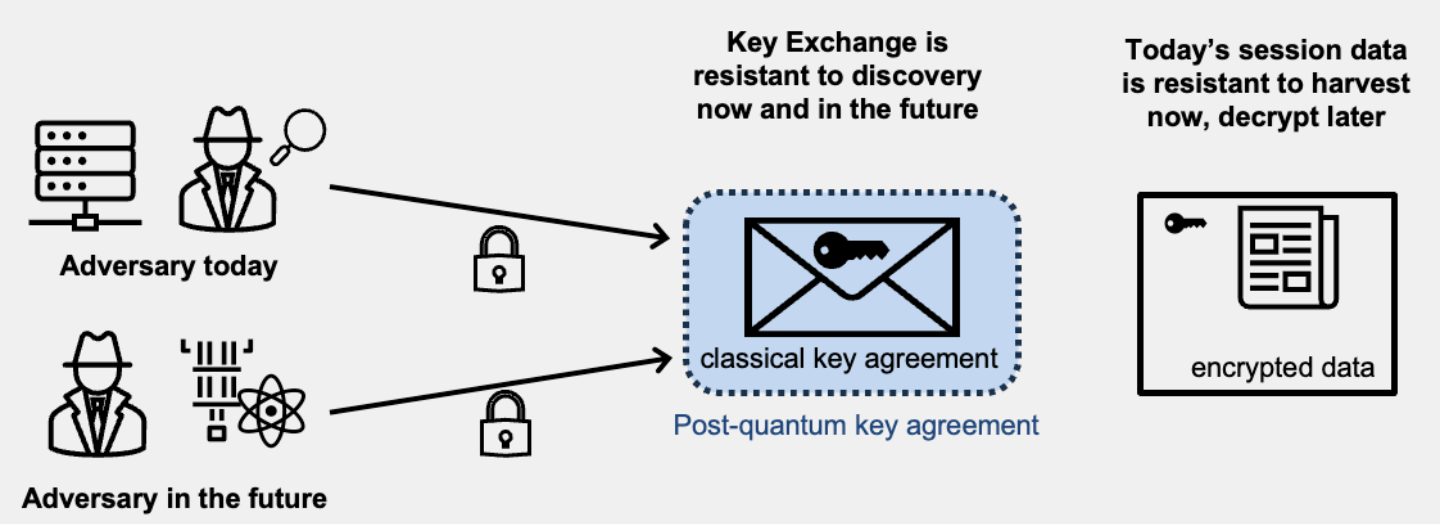
Les données que vous envoyez aux services activés sont protégées en transit par le chiffrement fourni par une connexion TLS (Transport Layer Security). Les suites de chiffrement classiques basées sur RSA et ECC prises en charge par AWS Payment Cryptography pour les sessions TLS rendent les attaques par force brute contre les mécanismes d'échange de clés irréalisables avec

la technologie actuelle. Toutefois, si les ordinateurs quantiques à grande échelle ou présentant un intérêt cryptographique (CRQC) deviennent pratiques à l'avenir, les mécanismes d'échange de clés TLS existants seront vulnérables à ces attaques. Il est possible que des adversaires commencent à récolter des données cryptées dès maintenant dans l'espoir de pouvoir les déchiffrer à l'avenir (récolter maintenant, déchiffrer plus tard). Si vous développez des applications qui reposent sur la confidentialité à long terme des données transmises via une connexion TLS, vous devriez envisager de passer à 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. AWS a mis en œuvre des suites de chiffrement hybrides à échange de clés post-quantique qui combinent des éléments classiques et post-quantiques pour 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 lorsque vous utilisez des versions récentes des kits SDK AWS. Pour plus d'informations sur la manière de enable/disable procéder à ce comportement, veuillez consulter ???



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

Les algorithmes AWS utilisés sont des algorithmes hybrides qui combinent Elliptic Curve Diffie-Hellman (ECDH), un algorithme d'échange de clés classique utilisé aujourd'hui dans le TLS, avec Module-Lattice-Based Key-Encapsulation Mechanism (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 clés post-quantique. Ce mécanisme hybride utilise chacun des algorithmes indépendamment pour générer une clé. Ensuite, il combine les deux clés cryptographiquement.

En savoir plus sur le PQC

Pour plus d'informations sur le projet de cryptographie post-quantique du National Institute for Standards and Technology (NIST), voir Cryptographie. Post-Quantum

Pour plus d'informations sur la normalisation de la cryptographie post-quantique par le NIST, voir Post-Quantum Standardisation de la cryptographie.

Activer le TLS post-quantique hybride

Les kits SDK et outils AWS possèdent des fonctionnalités et une configuration cryptographiques qui varient en fonction du langage et de l'environnement d'exécution. Un SDK ou un outil AWS fournit actuellement le support PQ TLS de trois manières :

Rubriques

- SDK avec PQ TLS activé par défaut

- Opt-in Prise en charge du protocole PQ TLS
- SDK basés sur le système OpenSSL
- Les kits SDK et outils AWS ne prévoient pas de prendre en charge le protocole PQ TLS

SDK avec PQ TLS activé par défaut

> **Note**
>
> À partir de 6Nov-2025, le SDK AWS et ses bibliothèques CRT sous-jacentes pour macOS et Windows utilisent des bibliothèques système pour TLS. Les fonctionnalités PQ TLS sur ces plateformes sont donc généralement déterminées par le support au niveau du système.

Kit AWS SDK pour Go

Le SDK AWS pour Go utilise la propre implémentation TLS de Golang fournie par sa bibliothèque standard. Golang prend en charge et préfère PQ TLS à partir de la version v1.24. Les utilisateurs du SDK AWS pour Go peuvent donc activer PQ TLS en mettant simplement à niveau Golang vers la version v1.24

SDK AWS pour JavaScript (navigateur)

Le SDK AWS pour JavaScript (navigateur) utilise la pile TLS du navigateur. Le SDK négociera donc le protocole TLS PQ si le moteur d'exécution du navigateur le prend en charge et le préfère. Firefox a lancé le support de PQ TLS dans la version 132.0. Chrome a annoncé la prise en charge de PQ TLS dans la version 131. Edge prend en charge le protocole PQ TLS opt-in dans la version 120 pour ordinateur de bureau et dans la version 140 pour Android.

SDK AWS pour Node.js

À partir de la Node.js version 22.20 (LTS) et de la version 24.9.0, OpenSSL 3.5 est Node.js lié et regroupe statiquement. Cela signifie que PQ TLS est activé et préféré par défaut pour ces versions et les suivantes.

Kit de développement logiciel AWS pour Kotlin

Le SDK Kotlin prend en charge et préfère PQ TLS sous Linux à partir de la v1.5.78. Étant donné que le SDK AWS pour le CRT-based client de Kotlin repose sur des bibliothèques système pour TLS sous

macOS et Windows, la prise en charge de PQ TLS dépendra de ces bibliothèques système sous-jacentes.

SDK AWS pour Rust

Le SDK AWS pour Rust distribue des packages distincts (appelés « caisses » dans l'écosystème Rust) pour chaque client de service. Ils sont tous gérés dans un GitHub référentiel consolidé, mais chaque client de service suit sa propre version et sa propre cadence de publication. Le SDK consolidé a publié la préférence PQ TLS sur 8/29 /25, de sorte que toute version de client de service individuelle publiée après cette date prendra en charge et préférera PQ TLS par défaut.

Vous pouvez déterminer la version minimale supportant le protocole PQ TLS pour un client de service donné en accédant à l'URL de la version de crates.io correspondante (par exemple, AWS Payment Cryptography se trouve ici) et en recherchant la première version publiée après 29-. Aug-25 Toute version du client de service publiée après le 29- Aug-25 aura PQ TLS activé et préféré par défaut.

Opt-in Prise en charge du protocole PQ TLS

Kit AWS SDK pour C++

Par défaut, le SDK C++ utilise des clients natifs de la plateforme tels que libcurl et. WinHttp Libcurl s'appuie généralement sur le système OpenSSL pour TLS, donc PQ TLS n'est activé par défaut que si le système OpenSSL est ≥ v3.5. Vous pouvez remplacer cette valeur par défaut dans le SDK C++ v1.11.673 ou version ultérieure, et opter pour le logiciel AwsCrtHttpClient qui prend en charge et active PQ TLS par défaut.

Remarques sur la création de TLS pour Opt-In PQ Vous pouvez récupérer les dépendances CRT du SDK à l'aide de ce script. La création du SDK à partir des sources est décrite ici et ici, mais notez que vous aurez peut-être besoin de quelques indicateurs CMake supplémentaires :

```
-DUSE_CRT_HTTP_CLIENT=ON \
-DUSE_TLS_V1_2=OFF \
-DUSE_TLS_V1_3=ON \
-DUSE_OPENSSL=OFF \
```

Kit AWS SDK pour Java

À partir de la version 2, le SDK AWS pour Java fournit un client HTTP AWS Common Runtime (AWS CRT) qui peut être configuré pour exécuter le protocole PQ TLS. Depuis la version 2.35.11, le AwsCrtHttpClient protocole TLS PQ est activé et préféré par défaut partout où il est utilisé.

SDK basés sur le système OpenSSL

Plusieurs kits SDK et outils AWS dépendent de la libcrypto/libssl bibliothèque du système pour TLS. La bibliothèque système la plus souvent utilisée est OpenSSL. OpenSSL a activé le support PQ TLS dans la version 3.5. Le moyen le plus simple de configurer ces SDK et outils pour PQ TLS est donc de les utiliser sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

Vous pouvez également configurer un conteneur Docker pour utiliser OpenSSL 3.5 afin d'activer PQ TLS sur n'importe quel système prenant en charge Docker. Voir Post-quantum TLS en Python pour un exemple de configuration pour Python.

AWS CLI

La prise en charge du protocole TLS par PQ avec le programme d'installation de la CLI AWS sera bientôt disponible. Pour l'activer immédiatement, vous pouvez utiliser d'autres programmes d'installation pour l'AWS CLI, qui varient en fonction du système d'exploitation, et peuvent activer PQ TLS.

Pour macOS, installez l'AWS CLI via Homebrew et assurez-vous que votre Homebrew-vended openSSL est mis à niveau vers la version 3.5+. Vous pouvez le faire avec « brew install openssl @3 .6 » et valider avec « brew list | grep openssl ».

Pour Ubuntu ou Debian Linux : assurez-vous qu'OpenSSL 3.5+ est installé sur la distribution Linux que vous utilisez en tant que système OpenSSL. Installez ensuite l'AWS CLI à l'aide d'apt ou de PyPI. Avec ces prérequis, la CLI AWS fournie par apt ou PyPI sera configurée pour négocier. PQ-TLS Pour obtenir des instructions étape par étape pour valider l'installation, consultez le référentiel github et le billet de blog qui l'accompagne.

AWS SDK pour PHP

Le SDK AWS pour PHP repose sur le libssl/libcrypto système. Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

Kit AWS SDK pour Python (Boto3)

Le SDK AWS pour Python (Boto3) repose sur le système. libssl/libcrypto Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

Kit AWS SDK pour Ruby

Le SDK AWS pour Ruby repose sur le libssl/libcrypto système. Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins.

Kit AWS SDK pour .NET

Sous Linux, le SDK AWS pour .NET repose sur le libssl/libcrypto système. Pour utiliser PQ TLS, utilisez ce SDK sur une distribution de système d'exploitation sur laquelle OpenSSL 3.5 est installé au moins. Sous Windows et macOS, PQ TLS est disponible à partir de .NET 10 et Windows 11. Sur macOS, la prise en charge du protocole TLS 1.3 (condition préalable au protocole PQ TLS) peut être activée en optant pour le protocole Apple, comme décrit ici. Network.framework En supposant une version .NET minimale de 10, PQ TLS devrait alors être activé.

Les kits SDK et outils AWS ne prévoient pas de prendre en charge le protocole PQ TLS

Il n'est actuellement pas prévu de prendre en charge les SDK et outils linguistiques suivants :

- SDK AWS pour SAP
- Kit de développement logiciel AWS pour Swift
- Outils AWS pour Windows PowerShell

Bonnes pratiques de sécurité pour la cryptographie des AWS paiements

AWS La cryptographie des paiements prend en charge de nombreuses fonctionnalités de sécurité intégrées ou que vous pouvez éventuellement implémenter pour améliorer la protection de vos clés de chiffrement et garantir qu'elles sont utilisées aux fins prévues, notamment des politiques IAM, un ensemble complet de clés de conditions de politique pour affiner vos politiques clés et vos politiques IAM et l'application intégrée des règles PCI PIN concernant les blocs de clés.

> **Important**
>
> Les directives générales fournies ne constituent pas une solution de sécurité complète. Étant donné que toutes les bonnes pratiques ne conviennent pas à toutes les situations, elles ne sont pas censées être prescriptives.

- Utilisation des clés et modes d'utilisation : La cryptographie des AWS paiements suit et applique les restrictions relatives à l'utilisation des clés et au mode d'utilisation, telles que décrites dans la spécification des blocs de clés d'échange de clés interopérables sécurisés ANSI X9 TR 31-2018 et conformément à l'exigence de sécurité PCI PIN 18-3. Cela limite la possibilité d'utiliser une seule clé à des fins multiples et lie cryptographiquement les métadonnées de la clé (telles que les opérations autorisées) au contenu de la clé lui-même. AWS La cryptographie des paiements applique automatiquement ces restrictions, de sorte qu'une clé de chiffrement (TR31_K0_KEY_ENCRYPTION_KEY) ne peut pas également être utilisée pour le déchiffrement des données. Pour plus d'informations, consultez Comprendre les principaux attributs de la clé AWS de cryptographie des paiements.
- Limitez le partage des clés symétriques : ne partagez que les clés symétriques (telles que les clés de chiffrement par code PIN ou les clés de chiffrement par clé) avec au plus une autre entité. S'il est nécessaire de transmettre des informations sensibles à d'autres entités ou partenaires, créez des clés supplémentaires. AWS La cryptographie des paiements n'expose jamais le contenu d'une clé symétrique ou d'une clé privée asymétrique en clair.
- Utilisez des alias ou des tags pour associer des clés à certains cas d'utilisation ou à certains partenaires : les alias peuvent être utilisés pour indiquer facilement le cas d'utilisation associé à une clé, par exemple alias/BIN_12345_CVK pour désigner une clé de vérification de carte associée au BIN 12345. Pour plus de flexibilité, pensez à créer des balises telles que bin=12345, use_case=acquiring, country=us, partner=foo. Les alias et les tags peuvent également être utilisés pour limiter l'accès, par exemple en renforçant les contrôles d'accès entre l'émission et l'acquisition des cas d'utilisation.
- Pratiquez l'accès le moins privilégié : l'IAM peut être utilisé pour limiter l'accès à la production aux systèmes plutôt qu'aux individus, par exemple en interdisant aux utilisateurs individuels de créer des clés ou d'exécuter des opérations cryptographiques. L'IAM peut également être utilisé pour limiter l'accès aux commandes et aux touches susceptibles de ne pas s'appliquer à votre cas d'utilisation, par exemple pour limiter la capacité de générer ou de valider des épingles pour un acquéreur. Une autre façon d'utiliser l'accès le moins privilégié consiste à limiter les opérations sensibles (telles que l'importation de clés) à des comptes de service spécifiques. Pour obtenir des

exemples, consultez AWS Exemples de politiques basées sur l'identité en matière de cryptographie des paiements.

Voir aussi

- Gestion des identités et des accès pour la cryptographie des AWS paiements
- Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM

Validation de conformité pour la cryptographie des AWS paiements

Comme pour les autres AWS services, les clients ont besoin d'une compréhension claire du modèle de responsabilité partagée en matière de sécurité et de conformité. En tant que service prenant spécifiquement en charge les paiements, la conformité aux normes PCI applicables est particulièrement importante à comprendre pour les clients de AWS Payment Cryptography. AWS Les évaluations PCI DSS et PCI 3DS incluent la cryptographie des paiements. AWS Des références au service peuvent figurer dans les guides de responsabilité partagée, disponibles auprès de AWS Artifact, pour ces rapports. Les évaluations de sécurité et de Point-to-Point chiffrement par code PIN PCI (P2PE) sont spécifiques à la cryptographie des AWS paiements.

Cette section fournit des informations sur l'état et l'étendue de la conformité du service, ainsi que des informations utiles pour planifier la sécurité PCI PIN et les évaluations PCI P2PE de vos applications.

Rubriques

Conformité du service

Des auditeurs tiers évaluent la sécurité et la conformité de la cryptographie des AWS paiements dans le cadre de multiples programmes de AWS conformité. Il s'agit notamment du SOC, du PCI et d'autres.

AWS La cryptographie des paiements a été évaluée pour plusieurs normes PCI en plus des normes PCI DSS et PCI 3DS. Il s'agit notamment de la sécurité par code PIN PCI (code PIN PCI) et du cryptage PCI Point-to-Point (P2PE). Consultez les AWS Artifact attestations et les guides de conformité disponibles.

Pour obtenir la liste des AWS services concernés par des programmes de conformité spécifiques, voir Services AWS concernés par programme de conformité . Pour obtenir des renseignements généraux, consultez Programmes de conformitéAWS .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Lorsque vous utilisez le chiffrement des AWS paiements, votre responsabilité en matière de conformité dépend de la sensibilité de vos données, des objectifs de conformité de votre entreprise et des lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité Guides 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 de conformité Ressources de : cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- Évaluation des ressources à l'aide des règles énoncées dans le guide du AWS Config développeur :AWS Configévalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub CSPM—Ce AWS service fournit une vue complète de l'état de votre sécurité interne, AWS ce qui vous permet de vérifier votre conformité aux normes et aux meilleures pratiques du secteur de la sécurité.

Planification de la conformité au code PIN

Ce guide décrit la documentation et les preuves dont vous aurez besoin pour préparer une évaluation du code PIN PCI de votre application de traitement du code PIN qui utilise la cryptographie des AWS paiements.

Comme pour Services AWS les autres normes de conformité, il est de votre responsabilité d'utiliser le service en toute sécurité, de configurer le contrôle d'accès et d'utiliser les paramètres de sécurité conformément aux exigences du code PIN PCI. Ce guide traitera de ces configurations lorsque cela est approprié pour répondre à une exigence.

Rubriques

- Sujets courants
- Portée de l'évaluation
- Opérations de traitement des transactions

Sujets courants

La migration des applications d'une connexion au HSM vers un service géré tel que la cryptographie des AWS paiements soulève des problèmes et des concepts courants pour les clients et leurs évaluateurs. Cette section fournit des informations pour clarifier la manière dont l'utilisation sécurisée du service répond à ces situations.

Rubriques

Responsabilité partagée

Les clients qui ont assumé l'entière responsabilité de la sécurité et de la conformité des applications restructureront leur conformité afin de tirer parti de la gestion des clés, des contrôles de sécurité et des fonctionnalités HSM gérées de AWS Payment Cryptography (« le service »). Cela transférera complètement certaines exigences AWS, comme en témoignent les évaluations tierces AWS de Payment Cryptography. Certaines exigences seront partagées entre l'application du client et le service. Une application est chargée de :

- Fournir des informations précises au service
- Utilisation des contrôles de sécurité conformément aux recommandations du service et aux exigences de sécurité du code PIN PCI
- Mise en œuvre des contrôles de sécurité requis à l'aide des outils fournis par le service

Les clients et leurs évaluateurs utiliseront des guides de responsabilité partagée et de mise en œuvre publiés avec des attestations de conformité AWS Artifact pour mettre en œuvre les contrôles et le suivi des contrôles, puis planifier et terminer les évaluations.

Configuration minimale du HSM

La norme de sécurité des données PCI, norme fondamentale pour les autres normes PCI, exige que tous les systèmes soient configurés avec le minimum de fonctionnalités nécessaires à leur fonctionnement. Les normes PCI PIN, P2PE et autres normes de solution appliquent cette exigence HSMs à la solution. HSMs doit uniquement activer les fonctions nécessaires à la solution.

AWS les services doivent être traités comme des systèmes et configurés pour les fonctionnalités minimales requises. La norme PCI DSS (Payment Card Industry Data Security Standard) v4.0 sur AWS recommande d'utiliser IAM pour configurer les fonctionnalités minimales de chaque service AWS utilisé par la solution. Cela vaut également pour la cryptographie des AWS paiements. Les politiques IAM permettent d'obtenir des autorisations précises pour restreindre les fonctions cryptographiques uniquement aux composants de l'application qui en dépendent.

Échange de clés entre le client et APC

Code PIN Les exigences de sécurité 8-4 et 15-2 exigent que les clés publiques soient authentifiées pour l'échange et le chargement des clés et que leur intégrité soit protégée. Pour le chargement à distance des POI, décrit fonctionnellement dans le ANSI/ASC X9 TR-34 et régi par l'annexe A du code PIN PCI, les clés publiques sont le plus souvent transmises sous forme de certificats signés par une autorité de certification conforme à l'annexe A2. Pour les échanges entre organisations, les clés publiques utilisent d'autres mécanismes d'authenticité et d'intégrité.

Toutes les interactions entre le client et AWS se font via AWS APIs, qui authentifie mutuellement chaque appel d'API et garantit l'intégrité des appels et des réponses à l'aide du protocole TLS. L'authentification de l'application client est gérée par AWS Identity and Access Management à l'aide de mécanismes tels que les jetons de sécurité et le SigV4. Les points de terminaison de l'API AWS sont authentifiés par le client à l'aide de l'authentification du serveur TLS, intégrée à AWS. SDKs Le protocole TLS garantit ensuite la confidentialité et l'intégrité de toutes les données transmises entre le client et chaque API AWS.

APC APIs GetParametersForImport et ImportKey implémentez un transfert de clé du client au service. Bien que l'autorité de certification (CA) fournie par ne GetParametersForImport soit pas conforme à l'annexe A2, elle est sécurisée et propre au compte. Bien que cette autorité de certification ne soit pas fiable pour la conformité aux exigences 8-4 et 15-2, elle fournit une vérification de l'intégrité de la clé importée. Vous pouvez également utiliser votre propre autorité de certification en tirant parti de l' GetCertificateSigningRequest API.

Les mécanismes qui fournissent l'authentification par clé publique et l'assurance de l'intégrité sont les suivants :

- Authentification fournie par l'authentification de l'API AWS
- L'intégrité de la clé est assurée par la fonction MAC du certificat fourni par GetParametersForImport, même si les informations d'identité contenues dans le certificat ne sont pas fiables. L'intégrité de la clé est également garantie par le MAC utilisé par TLS pour protéger la session entre le client et AWS.

Les certificats et blocs de clés fournis par APC sont conformes à l'annexe A1, qui spécifie les exigences relatives aux certificats et à la protection des clés par des méthodes asymétriques.

Portée de l'évaluation

La première étape de la planification de toute évaluation consiste à documenter la portée. Pour le code PIN PCI, le champ d'application concerne les systèmes et les processus qui protègent PINs, y compris la protection des clés cryptographiques et des dispositifs qui les protègent, à savoir les terminaux de paiement, également appelés points-of-interaction (POI) HSMs, et les autres dispositifs cryptographiques sécurisés (SCD).

Nous ne répondrons pas aux exigences dont vous assumez l'entière responsabilité, car celles-ci concernent des domaines extérieurs au champ d'application du service. Par exemple, configuration et approvisionnement de terminaux de paiement. Reportez-vous au guide de responsabilité partagée en matière de cryptographie des AWS paiements pour le code PIN PCI, disponible sur AWS Artifact

Rubriques

Responsabilité partagée

AWS Payment Cryptography est une organisation de cryptage et de support (ESO) et un fournisseur de services tiers (TPS) acquérant des codes PIN, tels que définis par le programme de sécurité des codes PIN de Visa et répertoriés dans le registre mondial des fournisseurs de services de Visa, sous la rubrique « Amazon Web Services, LLC ». Cela signifie que le service est autorisé par Visa à être utilisé par un VisaNet processeur tiers (VNP) acquérant un code PIN, un VisaNet processeur client acquérant un code PIN agissant en tant que fournisseur de services, et par d'autres fournisseurs de TPS et d'ESO sans qu'une évaluation supplémentaire soit nécessaire par les évaluateurs du code PIN du client (évaluateurs de code PIN qualifiés PCI ou PCI QPA).

Les autres marques de cartes ou fournisseurs de réseaux de paiement peuvent s'appuyer sur le programme de sécurité du code PIN Visa ou avoir leurs propres programmes. Contactez-nous AWS Support pour toute question concernant la conformité des services pour les autres programmes de réseaux de paiement.

AWS fournit l'attestation de conformité de sécurité PCI PIN (AOC) et le guide de responsabilité partagée pour la cryptographie des AWS paiements en. AWS Artifact Le recours à des fournisseurs de services pour le traitement du code PIN est courant depuis de nombreuses années, mais la norme de sécurité PCI PIN, jusqu'à la version 3.1, ne traite pas de la gestion des fournisseurs de services tiers. Le programme de sécurité du code PIN Visa ne le fait pas non plus. Le client QPA a suivi le modèle établi avec la norme PCI DSS AOC et le guide des responsabilités partagées, qui consiste à considérer la AWS« conformité » comme un test réussi pour déterminer les exigences applicables.

Schémas réseau de haut niveau

Le modèle de rapport PCI PIN exige : « Pour les entités engagées dans le traitement de transactions basées sur le code PIN, fournissez un schéma de réseau décrivant les flux de transactions basés sur le code PIN avec l'utilisation du type de clé associé. En outre, KIFs les entités engagées dans la distribution de clés à distance à l'aide de techniques asymétriques devraient fournir des flux de matériel de clé. »

AWS Payment Cryptography a décrit la structure de service interne pour notre évaluation du code PIN PCI. Vos diagrammes illustreront l'appel au service APIs pour le traitement du code PIN.

Exemple de schéma de réseau de haut niveau pour une application PIN utilisant la cryptographie des AWS paiements :

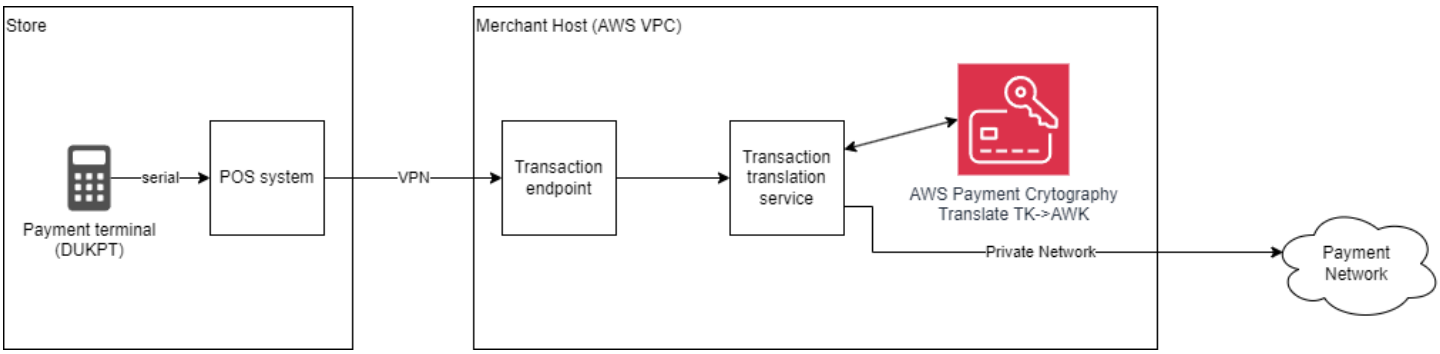


Table clé

Le rapport exige que toutes les clés protégeant PINs, directement ou indirectement, soient répertoriées. Toutes les clés présentes dans le service peuvent être répertoriées avec l'ListKeysAPI.

Assurez-vous de fournir la liste des clés de toutes les régions et de tous les comptes qui possèdent des clés pour votre application.

Références de documents

La documentation du fournisseur et les recommandations pour une utilisation sécurisée de la cryptographie des AWS paiements se trouvent dans le guide de l'utilisateur et le guide de référence de l'API. Ils sont liés, le cas échéant, dans ce guide.

Opérations de traitement des transactions

Les exigences relatives au code PIN PCI sont organisées dans les objectifs de contrôle. Chaque objectif de contrôle regroupe les exigences visant à garantir un aspect de la sécurité pour PINs.

Rubriques

Objectif de contrôle 1 : PINs les transactions régies par ces exigences sont traitées à l'aide d'équipements et de méthodologies garantissant leur sécurité.

Exigence 1 : les HSMs données utilisées par AWS Payment Cryptography ont été évaluées dans le cadre de notre évaluation du code PIN PCI. Pour les clients utilisant le service, les exigences 1 à 3 et 1 à 4 sont « en place » par rapport au HSM géré par le service. Les résultats pour HSM indiqueront que les tests ont été attestés par la AWS QPA. L'attestation de conformité PIN peut être consultée sur AWS Artifact. Les autres SCD de votre solution, tels que les POI, devront être inventoriés et référencés.

Exigence 2 : La documentation de vos procédures doit préciser comment les titulaires de carte PINs sont protégés en ce qui concerne la divulgation à votre personnel, le ou les protocoles de traduction du code PIN mis en œuvre et la protection lors du traitement en ligne et hors ligne. En outre, votre documentation doit contenir un résumé des méthodes de gestion des clés cryptographiques utilisées dans chaque zone.

Exigence 3 : le POI doit être configuré pour un cryptage et une transmission sécurisés par code PIN. AWS La cryptographie des paiements ne prend en charge que les traductions par blocs de code PIN spécifiées dans l'exigence 3-3.

Exigence 4 : L'application ne doit pas stocker de blocs de code PIN. Les blocs de code PIN, même chiffrés, ne doivent pas être conservés dans les journaux ou journaux de transactions. Le service ne stocke pas les blocs de code PIN et l'évaluation du code PIN vérifie qu'ils ne figurent pas dans les journaux.

Notez que la norme de sécurité PCI PIN s'applique à l'acquisition « de la gestion, du traitement et de la transmission sécurisés des données du numéro d'identification personnel (PIN) lors du traitement des transactions par carte de paiement en ligne ATMs et hors ligne sur les terminaux et point-of-sale (POS) », comme indiqué dans la norme. Cependant, la norme est souvent utilisée pour évaluer la gestion des clés cryptographiques pour les paiements en dehors de cette portée prévue. Cela peut inclure des cas d'utilisation par l'émetteur où PINs sont stockés. Les exceptions aux exigences applicables à ces cas doivent être convenues avec le public visé par l'évaluation.

Objectif de contrôle 2 : Les clés cryptographiques utilisées pour le code PIN encryption/decryption et la gestion des clés associées sont créées à l'aide de processus qui garantissent qu'il n'est pas possible de prédire une clé ou de déterminer que certaines clés sont plus probables que d'autres clés.

Exigence 5 : La génération de clés par cryptographie de AWS paiement a été évaluée dans le cadre de notre évaluation du code PIN PCI. Cela peut être spécifié dans la colonne « Généré par » du tableau clé.

Exigence 6 : Les contrôles de sécurité des clés détenues dans la cryptographie des AWS paiements ont été évalués dans le cadre de l'évaluation du code PIN PCI du service. Incluez des descriptions des contrôles de sécurité relatifs à la génération de clés au sein de votre application et auprès de tout autre fournisseur de services.

Exigence 7 : Vous devez disposer d'une documentation sur les politiques de génération de clés qui doit spécifier comment les clés sont générées et toutes les parties concernées doivent connaître

ces procédures/politiques. Les procédures de création de clés à l'aide de l'API APC doivent inclure l'utilisation de rôles dotés d'autorisations de création de clés et d'approbations pour exécuter des scripts ou tout autre code créant des clés. AWS CloudTrail les journaux contiennent tous les CreateKeyévénements avec la date et l'heure, l'ARN clé et les identifiants utilisateur. Les numéros de série HSM et les journaux d'accès aux supports physiques ont été évalués dans le cadre de l'évaluation du code PIN du service.

Objectif de contrôle 3 : Les clés sont transmises ou transmises de manière sécurisée.

Exigence 8 : La transmission des clés par cryptographie de AWS paiement a été évaluée dans le cadre de notre évaluation du code PIN PCI. Vous devrez documenter les mécanismes de protection des clés pour les transferts avant l'importation vers et après l'exportation depuis AWS Payment Cryptography. Le service fournit des valeurs de contrôle pour toutes les clés afin de valider le bon transport.

L'exigence 8-4 exige que les clés publiques soient transmises de manière à protéger leur intégrité et leur authenticité. Le transfert entre votre application et AWS est contrôlé par l'authentification de l'application AWS, à l'aide de Gestion des identités et des accès AWS méthodes, par l'authentification du point de AWS terminaison de l'API auprès de l'application via des certificats de serveur TLS. En outre, les clés publiques exportées ou importées vers AWS Payment Cryptography comportent des certificats signés de manière éphémère, spécifiques au client CAs (voir GetPublicKeyCertificate, et). GetParametersForImportGetParametersForExport Ils CAs ne peuvent pas être utilisés comme seule méthode d'authentification, car ils ne sont pas conformes à l'annexe A2 de sécurité du code PIN PCI. Cependant, les certificats garantissent toujours l'intégrité des clés publiques, IAM fournissant l'authentification.

Lorsque vous échangez des clés publiques avec vos partenaires commerciaux à l'aide de méthodes asymétriques, vous devez prévoir l'authentification de l'entreprise via le canal de communication, en utilisant un site Web d'échange de fichiers sécurisé, par exemple.

Exigence 9 : Le service n'utilise pas ou ne prend pas directement en charge les composants clés en texte clair.

Exigence 10 : Le service applique la force relative des clés en matière de protection des clés pour le transport. Vous êtes responsable du transfert des clés avant l'importation vers et après l'exportation depuis AWS Payment Cryptography et vous utilisez les paramètres de l'API et du TR-31 précis pour l'importation, l'exportation et la génération des clés. Vous devez disposer de procédures documentées décrivant les mécanismes de transfert des clés et la liste des clés cryptographiques utilisées pour le transport.

Exigence 11 : La documentation de vos procédures doit préciser le mode de transmission des clés. Les procédures de transfert de clés à l'aide de l'API de cryptographie des AWS paiements doivent inclure l'utilisation de rôles dotés d'autorisations d'importation et d'exportation de clés et d'approbations pour l'exécution de scripts ou d'autres codes créant des clés. AWS CloudTrail les journaux contiennent tous ImportKeyles ExportKeyévénements.

Objectif de contrôle 4 : Le chargement des clés vers les dispositifs d'acceptation du code PIN HSMs et des POI est géré de manière sécurisée.

Exigence 12 : Vous êtes responsable du chargement des clés à partir de composants ou de partages. La gestion des clés principales du HSM a été évaluée dans le cadre de l'évaluation du code PIN du service. AWS La cryptographie des paiements ne charge pas les clés provenant de partages ou de composants individuels. Consultez la section Détails cryptographiques.

Exigences 13 et 14 : Vous devrez décrire la protection clé pour les transferts avant et après l'importation depuis le service.

Exigence 15 : La cryptographie des AWS paiements fournit des valeurs de vérification des clés pour toutes les clés du service et une assurance d'intégrité pour les clés publiques. Votre application est chargée d'utiliser ces contrôles pour valider les clés après l'importation ou l'exportation depuis le service. Vous devez documenter les procédures pour vous assurer qu'un mécanisme de validation est en place.

L'exigence 15-2 exige que les clés publiques soient chargées de manière à protéger leur intégrité et leur authenticité. ImportKey, ainsi que GetParametersForImport, permet la validation des certificats de signature fournis. Si les certificats fournis sont auto-signés, l'authentification doit être fournie par un mécanisme distinct, par exemple un échange de fichiers sécurisé.

Exigence 16 : La documentation de vos procédures doit spécifier la manière dont les clés sont chargées dans le service. Les procédures d'importation de clés à l'aide de l'API doivent inclure l'utilisation de rôles dotés d'autorisations d'importation clés et d'approbations pour exécuter des scripts ou tout autre code chargeant des clés. AWS CloudTrail les journaux contiennent tous les ImportKeyévénements. Vous devez inclure les mécanismes de journalisation dans la documentation. Le service fournit des valeurs de vérification pour toutes les clés afin de valider le chargement correct des clés.

Objectif de contrôle 5 : Les clés sont utilisées de manière à empêcher ou à détecter leur utilisation non autorisée.

Exigence 17 : Le service fournit des mécanismes, tels que des balises et des alias, pour les clés qui permettent de suivre les relations de partage des clés. En outre, les valeurs de contrôle des touches doivent être conservées séparément pour démontrer que les valeurs clés connues ou par défaut ne sont pas utilisées lorsque les clés sont partagées.

Exigence 18 : Le service fournit des contrôles d'intégrité des clés, via GetKeyet ListKeys, et des événements de gestion des clés, via AWS CloudTrail, qui peuvent être utilisés pour détecter une substitution non autorisée ou surveiller la synchronisation des clés entre les parties. Le service stocke les clés exclusivement dans des blocs de clés. Vous êtes responsable du stockage et de l'utilisation des clés avant l'importation vers et après l'exportation depuis AWS Payment Cryptography.

Vous devez avoir mis en place des procédures pour une enquête immédiate en cas de divergence lors du traitement de transactions basées sur le code PIN ou d'événements imprévus liés à la gestion des clés.

Exigence 19 : Le service utilise des clés exclusivement dans les blocs clés, l'application KeyUsage et d'autres attributs clés pour toutes les opérations. KeyModeOfUse Cela inclut la restriction des opérations de clé privée. Vous devez utiliser vos clés publiques dans un seul but, par exemple pour le chiffrement ou la vérification de signature numérique, mais pas les deux. Vous devez utiliser des comptes distincts pour la production et test/development les systèmes.

Exigence 20 : Vous êtes responsable de cette exigence.

Objectif de contrôle 6 : Les clés sont administrées de manière sécurisée.

Exigence 21 : Le stockage des clés et leur utilisation avec la cryptographie des AWS paiements ont été évalués dans le cadre de l'évaluation du code PIN PCI du service. Pour les exigences de stockage liées aux composants clés, vous êtes responsable de les stocker comme indiqué aux points 21-2 et 21-3. Vous devrez décrire les principaux mécanismes de protection dans la documentation de votre politique avant l'importation vers le service et après l'exportation depuis celui-ci.

Exigence 22 : Les principales procédures de compromission pour la cryptographie des AWS paiements ont été évaluées dans le cadre de l'évaluation du code PIN PCI du service. Vous devrez décrire les principales procédures de détection des compromissions et de réponse, y compris la surveillance et la réponse aux notifications de AWS.

Exigence 23 : La cryptographie des AWS paiements ne prend pas en charge les variantes ni les autres méthodes de calcul de clés réversibles. Les clés principales APC ou les clés chiffrées par celles-ci ne sont jamais accessibles aux clients. L'utilisation du calcul des clés réversibles a été évaluée dans le cadre de l'évaluation du code PIN PCI du service.

Exigence 24 : Pratiques de destruction des secrets internes et des clés privées La cryptographie des AWS paiements a été évaluée dans le cadre de l'évaluation du code PIN PCI du service. Vous devrez décrire la procédure de destruction des clés avant l'importation vers et après l'exportation depuis APC. Les exigences relatives à la destruction des composants clés (24-2.2 et 24-2.3) restent à votre charge.

Exigence 25 : L'accès aux clés secrètes et privées dans le cadre de la cryptographie des AWS paiements a été évalué dans le cadre de l'évaluation du code PIN PCI du service. Vous aurez besoin d'un processus et d'une documentation pour les contrôles d'accès aux clés avant l'importation vers et après l'exportation depuis AWS Payment Cryptography.

Exigence 26 : Vous devez décrire la journalisation pour tout accès aux clés, aux composants clés ou au matériel connexe utilisé en dehors du service. Les journaux de toutes les activités de gestion clés effectuées par votre application avec le service sont disponibles via AWS CloudTrail.

Exigence 27 : Vous devrez décrire les procédures de sauvegarde des clés, des composants clés ou du matériel connexe utilisés en dehors du service.

Exigence 28 : Les procédures pour toute administration des clés utilisant l'API doivent inclure l'utilisation de rôles dotés d'autorisations d'administration clés et d'approbations pour l'exécution de scripts ou d'autres codes gérant les clés. AWS CloudTrail les journaux contiennent tous les événements d'administration clés

Objectif de contrôle 7 : L'équipement utilisé pour le traitement PINs et les clés est géré de manière sécurisée.

Exigence 29 : Vos exigences en matière de protections physiques et logiques HSMs sont satisfaites grâce à l'utilisation de la cryptographie des AWS paiements.

Exigence 30 : Votre application restera responsable de toutes les exigences relatives à la protection physique et logique des dispositifs POI.

Exigence 31 : La protection des dispositifs cryptographiques sécurisés (SCD) utilisés par AWS Payment Cryptography a été évaluée dans le cadre de l'évaluation du code PIN PCI du service. Vous devrez démontrer la protection de toute autre application SCDs utilisée par votre application.

Exigence 32 : L'utilisation de la cryptographie SCDs utilisée par les AWS paiements a été évaluée dans le cadre de l'évaluation du code PIN PCI du service. Vous devrez démontrer le contrôle d'accès et la protection de toute autre application SCDs utilisée par votre application.

Exigence 33 : Vous devrez décrire les protections de tout équipement de traitement PIN sous votre contrôle.

Utilisation du composant de déchiffrement AWS par cryptographie des paiements dans les solutions P2PE

Les solutions PCI P2PE peuvent utiliser le composant de déchiffrement par cryptographie des AWS paiements. Cela est documenté dans la section relative au Point-to-Point chiffrement PCI : exigences de sécurité et procédures de test, section Solutions P2PE et utilisation de tiers fournisseurs de composants and/or P2PE : « Un fournisseur de solutions (ou un commerçant en tant que fournisseur de solutions) peut sous-traiter certaines fonctions P2PE à des fournisseurs de composants P2PE répertoriés PCI et signaler l'utilisation des composants P2PE dans leur rapport de validation P2PE (P-ROV) », disponible sur le site Web de la PCI.

Comme pour les autres services AWS et normes de conformité, il est de votre responsabilité d'utiliser le service en toute sécurité, en configurant le contrôle d'accès et en utilisant des paramètres de sécurité conformes aux exigences de la norme PCI P2PE. Le guide de l'utilisateur du composant de déchiffrement P2PE d'AWS Payment Cryptography, disponible sur AWS Artifact, contient des instructions détaillées pour intégrer le chiffrement des AWS paiements à votre solution PCI P2PE et le rapport annuel sur les composants de déchiffrement, qui est requis pour les rapports de conformité.

Gestion des identités et des accès pour la cryptographie des AWS paiements

Gestion des identités et des accès AWS (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources de cryptographie des AWS paiements. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Rubriques

Public ciblé

La façon dont vous utilisez Gestion des identités et des accès AWS (IAM) varie en fonction de votre rôle :

- Utilisateur du service : demandez des autorisations à votre administrateur si vous ne pouvez pas accéder aux fonctionnalités (voir Résolution des problèmes d'identité et d'accès liés à la cryptographie des AWS paiements)
- Administrateur du service : déterminez l'accès des utilisateurs et soumettez les demandes d'autorisation (voir Comment fonctionne la cryptographie des AWS paiements avec IAM)
- Administrateur IAM : rédigez des politiques pour gérer l'accès (voir AWS Exemples de politiques basées sur l'identité en matière de cryptographie des paiements)

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 en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en tant qu'identité fédérée à l'aide d'informations d'identification provenant d'une source d'identité telle que AWS IAM Identity Center (IAM Identity Center), d'une authentification unique ou d'informations d'identification. Google/Facebook Pour plus d'informations sur la connexion, consultez Connexion à votre Compte AWS dans le Guide de l'utilisateur Connexion à AWS .

Pour l'accès par programmation, AWS fournit un SDK et une CLI pour signer les demandes de manière cryptographique. Pour plus d'informations, consultez Signature AWS Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une seule identité de connexion appelée utilisateur Compte AWS root qui dispose d'un accès complet à toutes Services AWS les ressources. Il est vivement déconseillé d'utiliser l'utilisateur racine pour vos tâches quotidiennes. Pour les tâches qui requièrent des informations d'identification de l'utilisateur racine, consultez Tâches qui requièrent les informations d'identification de l'utilisateur racine dans le Guide de l'utilisateur IAM.

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité qui dispose d'autorisations spécifiques pour une seule personne ou application. Nous vous recommandons d'utiliser ces informations d'identification temporaires au lieu des utilisateurs IAM avec des informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le guide de l'utilisateur IAM.

Les groupes IAM spécifient une collection d'utilisateurs IAM et permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. 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 dotée d'autorisations spécifiques qui fournit des informations d'identification temporaires. Vous pouvez assumer un rôle en passant d'un rôle d'utilisateur à un rôle

IAM (console) ou en appelant une opération d' AWS API AWS CLI ou d'API. Pour plus d'informations, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM sont utiles pour l'accès des utilisateurs fédérés, les autorisations temporaires des utilisateurs IAM, les accès intercompte, les accès entre services et les applications exécutées sur Amazon EC2. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Gestion de l'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 définit les autorisations lorsqu'elles sont associées à une identité ou à une ressource. AWS évalue ces politiques lorsqu'un directeur fait une demande. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations les documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

À l'aide de politiques, les administrateurs précisent qui a accès à quoi en définissant 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. Un administrateur IAM crée des politiques IAM et les ajoute aux rôles, que les utilisateurs peuvent ensuite assumer. Les politiques IAM définissent les autorisations quelle que soit la méthode que vous utilisez pour exécuter l'opération.

Identity-based politiques

Identity-based les politiques sont des documents de politique d'autorisation JSON que vous attachez à une identité (utilisateur, groupe ou rôle). Ces politiques contrôlent les actions que peuvent exécuter ces identités, 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.

Identity-based les politiques peuvent être des politiques intégrées (intégrées directement dans une seule identité) ou des politiques gérées (politiques autonomes associées à plusieurs identités). Pour découvrir comment choisir entre des politiques gérées et en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Resource-based politiques

Resource-based les politiques sont des documents de politique JSON que vous attachez à une ressource. Les exemples incluent les politiques de confiance de rôle IAM et les stratégies de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Vous devez spécifier un principal dans une politique basée sur les ressources.

Resource-based les politiques sont des politiques intégrées qui se trouvent dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d'accès (ACL)

Les listes de contrôle d'accès (ACL) vérifie quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et Amazon VPC sont des exemples de services qui prennent en charge les ACL. AWS WAF Pour en savoir plus sur les listes de contrôle d'accès, consultez Vue d'ensemble des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge des types de politiques supplémentaires qui peuvent définir les autorisations maximales accordées par les types de politiques les plus courants :

- Limites d'autorisations : une limite des autorisations définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCP) : spécifient les autorisations maximales pour une organisation ou une unité organisationnelle dans AWS Organizations. Pour plus d'informations, consultez Politiques de contrôle de service dans le Guide de l'utilisateur AWS Organizations .
- Politiques de contrôle des ressources (RCP) : définissent les autorisations maximales disponibles pour les ressources de votre organisation. Pour plus d'informations, consultez Politiques de contrôle des ressources (RCP) dans le Guide de l'utilisateur AWS Organizations .

- Politiques de session : politiques avancées que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. 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.

Comment fonctionne la cryptographie des AWS paiements avec IAM

Avant d'utiliser IAM pour gérer l'accès à la cryptographie des AWS paiements, vous devez comprendre quelles fonctionnalités IAM peuvent être utilisées avec AWS la cryptographie des paiements. Pour obtenir une vue d'ensemble du fonctionnement de la cryptographie des AWS paiements et AWS des autres services avec IAM, consultez la section AWS Services qui fonctionnent avec IAM dans le guide de l'utilisateur d'IAM.

Rubriques

- AWS Politiques de cryptographie Identity-based des paiements
- Autorisation basée sur les balises AWS de cryptographie des paiements

AWS Politiques de cryptographie Identity-based des paiements

Avec les politiques basées sur l'identité IAM, vous pouvez spécifier les actions et les ressources autorisées ou refusées ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. AWS La cryptographie des paiements prend en charge des actions, des ressources et des clés de condition spécifiques. Pour en savoir plus sur tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l’utilisateur IAM.

Actions

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Les actions politiques dans AWS Payment Cryptography utilisent le préfixe suivant avant l'action :. `payment-cryptography:` Par exemple, pour autoriser une personne à exécuter une opération d'`VerifyCardData`API de chiffrement des AWS paiements, vous devez inclure cette `payment-cryptography:VerifyCardData` action dans sa politique. Les déclarations de politique doivent inclure un élément `Action` ou `NotAction`. AWS La cryptographie des paiements définit son propre ensemble d'actions décrivant les tâches que vous pouvez effectuer avec ce service.

Pour spécifier plusieurs actions dans une seule déclaration, séparez-les par des virgules comme suit :

```
"Action": [
      "payment-cryptography:action1",
      "payment-cryptography:action2"
```

Vous pouvez aussi préciser plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `List` (telles que `ListKeys` et`ListAliases`), incluez l'action suivante :

```
"Action": "payment-cryptography:List*"
```

Pour consulter la liste des actions de cryptographie des AWS paiements, consultez la section Actions définies par la cryptographie des AWS paiements dans le guide de l'utilisateur IAM.

Ressources

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

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

La ressource clé de cryptographie de paiement possède l'ARN suivant :

```
arn:${Partition}:payment-cryptography:${Region}:${Account}:key/${keyARN}
```

Pour plus d'informations sur le format des ARN, consultez les sections Amazon Resource Names (ARN) et AWS Service Namespaces.

Par exemple, pour spécifier l'instance `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` dans votre instruction, utilisez l'ARN suivant :

```
"Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h"
```

Pour spécifier toutes les clés appartenant à un compte spécifique, utilisez le caractère générique (*) :

```
"Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/*"
```

Certaines actions AWS de chiffrement des paiements, telles que celles relatives à la création de clés, ne peuvent pas être effectuées sur une ressource spécifique. Dans ces cas-là, vous devez utiliser le caractère générique (*).

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

Pour spécifier plusieurs ressources dans une seule instruction, utilisez une virgule comme indiqué ci-dessous :

```
"Resource": [
      "resource1",
      "resource2"
```

Exemples

Pour consulter des exemples de politiques basées sur l'identité en matière de cryptographie des AWS paiements, consultez. AWS Exemples de politiques basées sur l'identité en matière de cryptographie des paiements

Autorisation basée sur les balises AWS de cryptographie des paiements

Vous pouvez associer des balises aux ressources de cryptographie des AWS paiements ou transmettre des balises dans une demande adressée à AWS Payment Cryptography. Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `payment-cryptography:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`.

AWS Exemples de politiques basées sur l'identité en matière de cryptographie des paiements

Par défaut, les utilisateurs et les rôles IAM ne sont pas autorisés à créer ou à modifier des ressources de chiffrement des AWS paiements. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l' AWS API AWS Management Console AWS CLI, ou. Un administrateur IAM doit créer des politiques IAM autorisant les utilisateurs et les rôles à exécuter des opérations d'API spécifiques sur les ressources spécifiées dont ils ont besoin. Il doit ensuite attacher ces politiques aux utilisateurs ou aux groupes IAM ayant besoin de ces autorisations.

Pour savoir comment créer une stratégie IAM basée sur l'identité à l'aide de ces exemples de documents de stratégie JSON, veuillez consulter Création de stratégies dans l'onglet JSON dans le Guide de l'utilisateur IAM.

Rubriques

- Bonnes pratiques en matière de politiques
- Utilisation de la console AWS de cryptographie des paiements
- Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations
- Possibilité d'accéder à tous les aspects de la cryptographie des AWS paiements
- Possibilité d'appeler des API à l'aide de touches spécifiées
- Possibilité de refuser spécifiquement une ressource

Bonnes pratiques en matière de politiques

Identity-based les politiques déterminent si quelqu'un peut créer, accéder ou supprimer des ressources AWS de chiffrement des paiements dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue.

Compte AWS Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la console AWS de cryptographie des paiements

Pour accéder à la console AWS Payment Cryptography, vous devez disposer d'un ensemble minimal d'autorisations. Ces autorisations doivent vous permettre de répertorier et de consulter les informations relatives aux ressources AWS de chiffrement des paiements de votre AWS compte. Si vous créez une politique basée sur l'identité qui est plus restrictive que les autorisations minimales requises, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs et rôles IAM) tributaires de cette politique.

Pour garantir que ces entités peuvent toujours utiliser la console AWS Payment Cryptography, associez également la politique AWS gérée suivante aux entités. Pour plus d'informations, consultez Ajout d'autorisations à un utilisateur dans le Guide de l'utilisateur IAM.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement l'API AWS CLI ou l' AWS API. Autorisez plutôt l'accès à uniquement aux actions qui correspondent à l'opération d'API que vous tentez d'effectuer.

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

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

```
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Possibilité d'accéder à tous les aspects de la cryptographie des AWS paiements

> **Warning**
>
> Cet exemple fournit des autorisations étendues et n'est pas recommandé. Envisagez plutôt les modèles d'accès les moins privilégiés.

Dans cet exemple, vous souhaitez accorder à un utilisateur IAM de votre AWS compte l'accès à toutes vos clés de chiffrement des AWS paiements et la possibilité d'appeler toutes les API de cryptographie des AWS paiements, y compris les deux `ControlPlane` opérations. `DataPlane`

JSON

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

Possibilité d'appeler des API à l'aide de touches spécifiées

Dans cet exemple, vous souhaitez accorder à un utilisateur IAM de votre AWS compte l'accès à l'une de vos clés de chiffrement des AWS paiements, `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` puis utiliser cette ressource dans deux API, `GenerateCardValidationData` et. `VerifyCardValidationData` À l'inverse, l'utilisateur IAM n'aura pas accès à cette clé pour d'autres opérations telles que `DeleteKey` ou `ExportKey`

Les ressources peuvent être soit des clés, préfixées par, `key` soit des alias, préfixés par. `alias`

JSON

```
   {
      "Version":"2012-10-17",
      "Statement": [
            {
               "Effect": "Allow",
               "Action": [
                  "payment-cryptography:VerifyCardValidationData",
                  "payment-cryptography:GenerateCardValidationData"
               ],
```

```
                "Resource": [
                   "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
                ]
            }
        ]
    }
```

Possibilité de refuser spécifiquement une ressource

> **Warning**
>
> Réfléchissez bien aux implications de l'octroi d'un accès par caractère générique. Envisagez plutôt un modèle de moindre privilège.

Dans cet exemple, vous souhaitez autoriser un utilisateur IAM de votre AWS compte à accéder à n'importe laquelle de vos clés de chiffrement des AWS paiements, mais vous souhaitez refuser les autorisations relatives à une clé spécifique. L'utilisateur aura accès à `VerifyCardData` et `GenerateCardData` avec toutes les clés, à l'exception de celle spécifiée dans la déclaration de refus.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
              "Action": [
                  "payment-cryptography:VerifyCardValidationData",
                  "payment-cryptography:GenerateCardValidationData"
              ],
              "Resource": [
                "arn:aws:payment-cryptography:us-east-2:111122223333:key/*"
              ]
          },
          {
              "Effect": "Deny",
```

```
            "Action": [
              "payment-cryptography:GenerateCardValidationData"
            ],
            "Resource": [
              "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h"
            ]
          }
      ]
  }
```

Resource-based politiques relatives à la cryptographie des AWS paiements

Resource-based les politiques sont des documents de politique JSON que vous attachez à une ressource, comme une clé AWS de chiffrement des paiements. Dans une politique basée sur les ressources, vous spécifiez qui peut accéder à la clé et les actions qu'il peut effectuer sur celle-ci. Vous pouvez utiliser des stratégies basées sur une ressource pour :

- Accordez l'accès à une clé unique à plusieurs utilisateurs et rôles.
- Accordez l'accès à des utilisateurs ou à des rôles dans d'autres AWS comptes.

Rubriques

Lorsque vous associez une politique basée sur les ressources à une clé de chiffrement des AWS paiements, la cryptographie des AWS paiements utilise la logique d'évaluation de la politique IAM pour déterminer si un principal donné est autorisé à effectuer l'action demandée. Pour activer l'accès entre comptes, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte comme principal dans une politique basée sur les ressources. Cross-account l'accès nécessite deux politiques :

1. Resource-based politique (compte du propriétaire de la clé) — Le propriétaire de la clé utilise `PutResourcePolicy` pour accorder l'accès au compte de l'appelant ou au principal IAM.

2. Identity-based politique (compte de l'appelant) — L'administrateur IAM de l'appelant doit également autoriser l'action de cryptographie des AWS paiements (par exemple,`payment-cryptography:EncryptData`) dans la politique IAM de l'appelant.

Les deux politiques doivent autoriser l'action. Si l'un ou l'autre est manquant, la demande multi-comptes est refusée avec`AccessDeniedException`.

Si une politique basée sur les ressources accorde l'accès à un mandant du même compte, aucune politique supplémentaire basée sur l'identité n'est requise. Pour plus d'informations, consultez la section En quoi les rôles IAM diffèrent des Resource-based politiques dans le guide de l'utilisateur IAM.

⚠ Opérations du plan de contrôle des politiques de ressources

Resource-based les politiques ne s'appliquent pas aux opérations du plan de contrôle des politiques de ressources telles que `PutResourcePolicy``GetResourcePolicy`, et `DeleteResourcePolicy`. Cela permet d'éviter les scénarios de verrouillage potentiels dans lesquels une politique de ressources pourrait refuser la possibilité de modifier ou de supprimer la politique elle-même. L'accès à ces opérations du plan de contrôle est régi uniquement par les politiques basées sur l'identité IAM.

Considérations

Gardez les points suivants à l'esprit lorsque vous utilisez des politiques basées sur les ressources avec AWS Payment Cryptography.

- AWS La cryptographie des paiements n'impose automatiquement aucun accès public aux clés. Vous ne pouvez pas créer de politique basée sur les ressources qui accorde l'accès à des personnes anonymes ou publiques. Tout accès aux clés AWS de cryptographie de paiement nécessite des AWS principes authentifiés, et l'accès public est toujours bloqué.
- Resource-based les politiques sont appliquées par clé. Chaque clé AWS de cryptographie de paiement peut être associée à au plus une politique basée sur les ressources.
- Resource-based les politiques ne s'appliquent pas aux alias. Lorsque vous référencez une clé par son alias, la politique de ressources attachée à la clé sous-jacente est évaluée.

- Resource-based les politiques ne s'appliquent pas aux clés de région de réplication en lecture seule créées à l'aide de la réplication de Multi-Region clés pour le moment. Les politiques relatives aux ressources ne peuvent être associées qu'à la clé de région principale.
- L'`Resource`élément d'une politique basée sur les ressources doit être `"*"` ou correspondre exactement à l'ARN de la clé à laquelle la politique est attachée. Son utilisation `"*"` est recommandée car elle permet de réutiliser le même document de politique sur plusieurs clés.
- Les API de gestion des politiques de ressources (`PutResourcePolicy`,`GetResourcePolicy`, et`DeleteResourcePolicy`) sont limitées à Compte AWS celles qui possèdent la clé. Seuls les principaux titulaires du compte du propriétaire de la clé peuvent gérer les politiques relatives aux ressources.

Gestion des politiques basées sur les ressources

Vous pouvez gérer les politiques basées sur les ressources pour les clés de chiffrement des AWS paiements à l'aide de l' AWS CLI API or. AWS Pour utiliser cette commande, remplacez celle *`italicized placeholder text`* de l'exemple par vos propres informations.

Joindre une politique basée sur les ressources

Utilisez l'action `PutResourcePolicy`API ou la commande put-resource-policyCLI pour associer une politique basée sur les ressources à une clé. Si une politique existe déjà, la commande la remplace.

L'exemple suivant associe une politique basée sur les ressources d'un fichier JSON à une clé.

```
aws payment-cryptography put-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --policy file://policy.json
```

Récupérer une politique basée sur les ressources

Utilisez l'action `GetResourcePolicy`API ou la commande get-resource-policyCLI pour récupérer la politique basée sur les ressources attachée à une clé.

L'exemple suivant récupère la politique basée sur les ressources attachée à une clé.

```
aws payment-cryptography get-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

La réponse renvoie le document de politique :

```
{
    "Policy": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
                },
                "Action": [
                    "payment-cryptography:EncryptData",
                    "payment-cryptography:DecryptData"
                ],
                "Resource": "*"
            }
        ]
    }
}
```

Supprimer une politique basée sur les ressources

Utilisez l'action DeleteResourcePolicyAPI ou la commande delete-resource-policyCLI pour supprimer la politique basée sur les ressources d'une clé.

L'exemple suivant supprime la politique basée sur les ressources attachée à une clé.

```
aws payment-cryptography delete-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

Resource-based exemples de politiques

Accorder l'accès à une clé entre comptes

La politique basée sur les ressources suivante accorde à un rôle dans un autre AWS compte l'autorisation d'utiliser une clé de cryptographie AWS de paiement pour des opérations cryptographiques.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
            },
            "Action": [
                "payment-cryptography:GenerateCardValidationData",
                "payment-cryptography:VerifyCardValidationData"
            ],
            "Resource": "*"
        }
    ]
}
```

Accorder différentes autorisations à différents comptes

La politique basée sur les ressources suivante montre comment accorder différentes autorisations aux principaux dans des comptes distincts. Dans cet exemple, un serveur de contrôle d'accès 3DS (ACS) d'un compte peut générer des données de validation de carte, tandis qu'un service d'autorisation de paiement d'un autre compte ne peut valider que les cryptogrammes 3DS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow3DSACSToGenerate",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/3dsAcsRole"
            },
            "Action": [
                "payment-cryptography:GenerateCardValidationData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowPaymentAuthToVerify",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::444455556666:role/PaymentAuthRole"
            },
```

```
            "Action": [
                "payment-cryptography:VerifyAuthRequestCryptogram"
            ],
            "Resource": "*"
        }
    ]
}
```

Multi-party approbation de la cryptographie des AWS paiements

AWS La cryptographie des paiements s'intègre à Multi-party l'approbation (MPA), une fonctionnalité d'Amazon Web Services Organizations, pour aider à protéger les opérations critiques grâce à un processus d'approbation distribué. Avec MPA, vous pouvez exiger que plusieurs personnes de confiance approuvent des opérations de cryptographie de AWS paiement spécifiques avant qu'elles ne soient effectuées.

Rubriques

Présentation de

Multi-party L'approbation ajoute un niveau de sécurité supplémentaire aux opérations sensibles de cryptographie des AWS paiements en exigeant l'approbation d'un groupe de personnes de confiance avant que l'opération ne puisse se poursuivre. Cela permet de se protéger contre les modifications non autorisées si un seul ensemble d'informations d'identification est compromis et d'empêcher une seule personne d'apporter une modification unilatérale.

Une équipe d'approbation est un groupe d'approbateurs au sein de votre organisation que vous désignez pour approuver ou refuser les demandes d'opérations protégées. Le processus

d'approbation est entièrement géré par les approbateurs de votre organisation. Aucun AWS membre du personnel n'est impliqué dans l'approbation ou le refus des demandes.

Lorsque le MPA est activé pour une opération protégée, les événements suivants se produisent :

1. Un demandeur lance l'opération protégée.
2. MPA crée une session d'approbation et en informe les membres de l'équipe d'approbation.
3. Les membres de l'équipe d'approbation examinent la demande et l'approuvent ou la rejettent via le portail MPA.
4. Une fois que le seuil minimum d'approbations requis est atteint, l'opération se poursuit. Si la demande est refusée par l'équipe d'approbation ou si la durée de session autorisée expire avant que le seuil d'approbation ne soit atteint, l'opération n'est pas effectuée. Dans les deux cas, le demandeur doit soumettre une nouvelle demande pour réessayer l'opération.

> **Note**
>
> Lors de l'importation d'un certificat CA racine avec MPA activé, le `RequesterComment` paramètre est obligatoire. Ce commentaire est inclus dans la notification d'approbation envoyée à l'équipe d'approbation, fournissant le contexte de la demande.

Opérations protégées

AWS Payment Cryptography prend en charge le MPA pour les opérations suivantes :

- `ImportKey`avec du matériel `RootCertificatePublicKey` clé — L'importation d'un certificat de clé publique racine est une opération critique car les certificats racine constituent l'ancre de confiance pour toutes les importations et exportations de clés ultérieures à l'aide d'un échange de clés asymétrique tel que TR-34. Le fait d'exiger l'approbation de plusieurs parties pour cette opération permet de garantir qu'aucune personne ne puisse établir ou modifier unilatéralement le fondement de la confiance en ce qui concerne vos clés de cryptographie de AWS paiement.

Conditions préalables

Avant de pouvoir utiliser MPA avec AWS le chiffrement des paiements, vous devez remplir les conditions préalables suivantes :

- Configurez MPA dans votre environnement Amazon Web Services Organizations. Pour obtenir des instructions, voir Qu'est-ce que Multi-party l'approbation ? dans le guide Multi-party d'utilisation relatif à l'approbation.
- Créez au moins une équipe d'approbation avec les approbateurs requis.
- Partagez l'équipe d'approbation avec Compte AWS celle qui contient vos clés de cryptographie AWS de paiement à l'aide AWS Resource Access Manager de.
- Le compte de gestion de votre organisation doit être activé pour être Multi-party approuvé.

Activation et désactivation de MPA

Après avoir mis en place une équipe d'approbation, vous pouvez activer MPA pour le chiffrement des AWS paiements en associant l'équipe à votre compte. Vous pouvez également désactiver le MPA en dissociant l'équipe, bien que la dissociation nécessite l'approbation de l'équipe d'approbation actuellement associée.

Activer MPA

Utilisez l'action `AssociateMpaTeam` API ou la commande associate-mpa-team CLI pour associer une équipe d'approbation à votre compte AWS Payment Cryptography. Une fois associées, les opérations protégées nécessitent l'approbation de l'équipe avant de pouvoir être poursuivies.

```
aws payment-cryptography associate-mpa-team \
    --team-arn arn:aws:mpa:us-east-1:111122223333:team/my-approval-team
```

Désactiver MPA

Utilisez l'action `DisassociateMpaTeam` API ou la commande disassociate-mpa-team CLI pour supprimer l'association de l'équipe d'approbation. Dissocier une équipe est en soi une opération protégée qui nécessite l'approbation de l'équipe d'approbation actuellement associée.

```
aws payment-cryptography disassociate-mpa-team \
    --team-arn arn:aws:mpa:us-east-1:111122223333:team/my-approval-team
```

> **Important**
>
> La désactivation de MPA nécessite l'approbation de l'équipe d'approbation actuellement associée. Cela garantit qu'aucune personne ne peut supprimer unilatéralement la protection d'approbation multipartite.

> **Note**
>
> Le `--requester-comment` paramètre est facultatif pour associate-mpa-team etdisassociate-mpa-team.

Prise en main

Pour commencer à utiliser MPA for AWS Payment Cryptography, consultez le guide de Multi-party l'utilisateur d'approbation pour obtenir des instructions de configuration détaillées, notamment comment créer des équipes d'approbation, configurer les politiques d'approbation et gérer les sessions d'approbation.

Exemple : Importer un certificat racine avec MPA activé

Une fois que le MPA est activé et qu'une équipe d'approbation est associée à l'`ImportKey`opération`RootCertificatePublicKey`, la demande d'importation doit être approuvée avant de pouvoir être traitée.

1. Un demandeur appelle `import-key` pour importer un certificat de clé publique racine. Pour utiliser cette commande, remplacez celle *`italicized placeholder text`* de l'exemple par vos propres informations.

```
aws payment-cryptography import-key \
    --key-material='{"RootCertificatePublicKey": {
    "KeyAttributes": {
        "KeyAlgorithm": "RSA_4096",
        "KeyClass": "PUBLIC_KEY",
        "KeyModesOfUse": {"Verify": true},
        "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
    },
    "PublicKeyCertificate": "LS0tLS1CRUdJTi..."}}' \
```

```
    --requester-comment "Importing new root CA certificate for TR-34 key exchange
 with partner XYZ"
```

La réponse renvoie une clé `KeyState` définie sur`CREATE_IN_PROGRESS`, indiquant que la demande est en attente d'approbation. La réponse inclut également `MpaStatus` des détails sur la session d'approbation :

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_IN_PROGRESS",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-
east-1:111122223333:session/abc123def456",
            "Status": "PENDING",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00"
        }
    }
}
```

2. Le MPA étant activé, la demande ne se termine pas immédiatement. AWS Payment Cryptography crée plutôt une session d'approbation et renvoie une réponse indiquant que l'approbation est en attente.
3. Les membres de l'équipe d'approbation reçoivent une notification et examinent la demande via le portail MPA. Une fois que le nombre requis d'approbateurs a approuvé la demande, l'opération d'importation se poursuit et le certificat racine est importé.

AWS CloudTrail journalisation des événements MPA

Lorsque MPA est activé, AWS Payment Cryptography enregistre les événements du service AWS CloudTraillorsqu'une session d'approbation est terminée. Ces événements enregistrent le résultat du processus d'approbation, notamment si la demande a été approuvée ou a échoué. Vous pouvez utiliser ces journaux pour auditer l'activité des MPA et suivre l'état des opérations protégées.

MPA-related CloudTrail les événements incluent les champs suivants dans `serviceEventDetails` :

- `keyArn`— L'ARN de la clé affectée par l'opération.
- `operation`— L'opération protégée qui a été demandée.
- `mpaSessionArn`— L'ARN de la session d'approbation du MPA.
- `sessionStatus`— Le résultat de la session d'approbation (`APPROVED`ou`FAILED`).

Demande approuvée

L'exemple suivant montre un CloudTrail événement lié à une `ImportKey` demande approuvée par l'équipe MPA :

```
{
    "eventVersion": "1.11",
    "eventTime": "2026-04-28T18:49:51Z",
    "eventName": "ImportKey",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventType": "AwsServiceEvent",
    "eventCategory": "Management",
    "awsRegion": "us-east-1",
    "readOnly": false,
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "payment-cryptography.amazonaws.com"
    },
    "resources": [
        {
            "ARN": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/spa2dclzmsihlj4o",
            "accountId": "111122223333",
```

```
            "type": "AWS::PaymentCryptography::Key"
        }
    ],
    "serviceEventDetails": {
        "keyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/spa2dclzmsihlj4o",
        "operation": "ImportKey",
        "mpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/my-approval-
team/44c76e07-8937-4d7d-bb9a-a646322e2a1e",
        "sessionStatus": "APPROVED"
    }
}
```

Demande échouée

L'exemple suivant montre un CloudTrail événement lié à une `ImportKey` demande refusée ou dont le délai a expiré :

```
{
    "eventVersion": "1.11",
    "eventTime": "2026-04-28T18:50:35Z",
    "eventName": "ImportKey",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventType": "AwsServiceEvent",
    "eventCategory": "Management",
    "awsRegion": "us-east-1",
    "readOnly": false,
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "payment-cryptography.amazonaws.com"
    },
    "resources": [
        {
            "ARN": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/qj46ku4qimypxdo7",
            "accountId": "111122223333",
            "type": "AWS::PaymentCryptography::Key"
        }
    ],
    "serviceEventDetails": {
```

```
        "keyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/qj46ku4qimypxdo7",
        "operation": "ImportKey",
        "mpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/my-approval-team/
b0ac1994-14e1-47a6-bf1a-0b6fc0b845f2",
        "sessionStatus": "FAILED"
    }
}
```

Pour en savoir plus AWS CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Vérification de l'état des demandes et gestion des échecs

Vous pouvez vérifier l'état d'une demande MPA en attente en appelant `GetKey`. La réponse inclut le `MpaStatus` champ contenant les détails de la session d'approbation en cours. Pour utiliser cette commande, remplacez celle *`italicized placeholder text`* de l'exemple par vos propres informations.

```
aws payment-cryptography get-key \
    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

Pendant que la demande est en attente d'approbation, la réponse s'affiche `KeyState` sous forme `CREATE_IN_PROGRESS` et `MpaStatus.Status` sous la forme suivante `PENDING` :

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_IN_PROGRESS",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/abc123def456",
            "Status": "PENDING",
```

```
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00"
        }
    }
}
```

Une fois que le nombre requis d'approbateurs a approuvé la demande, celle-ci est `KeyState` transférée `CREATE_COMPLETE` et `MpaStatus.Status` déplacée vers. `APPROVED` La clé est maintenant prête à être utilisée :

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/abc123def456",
            "Status": "APPROVED",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00"
        }
    }
}
```

Si la demande est refusée par l'équipe d'approbation ou si la session expire avant que le seuil d'approbation ne soit atteint, les `KeyState` modifications apportées `CREATE_FAILED` sont `MpaStatus.Status` les `FAILED` suivantes :

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
```

```
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_FAILED",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/abc123def456",
            "Status": "FAILED",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00",
            "StatusMessage": "Approval session expired or was denied"
        }
    }
}
```

Une clé dont le `CREATE_FAILED` statut est défini ne peut pas être utilisée pour des opérations cryptographiques. Pour réessayer l'importation, vous devez soumettre une nouvelle `ImportKey` demande, ce qui créera une nouvelle session d'approbation.

Résolution des problèmes d'identité et d'accès liés à la cryptographie des AWS paiements

Des sujets seront ajoutés à cette section au fur et à mesure que les IAM-related problèmes spécifiques à la cryptographie des AWS paiements seront identifiés. Pour obtenir du contenu général sur le dépannage sur des sujets liés à l'IAM, reportez-vous à la section de résolution des problèmes du guide de l'utilisateur IAM.

Surveillance de la cryptographie des AWS paiements

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances de AWS Payment Cryptography et de vos autres solutions AWS. AWS fournit les outils de surveillance suivants pour surveiller le chiffrement des AWS paiements, signaler tout problème et prendre des mesures automatiques le cas échéant :

- Amazon CloudWatch surveille vos AWS ressources et les applications que vous utilisez AWS en temps réel. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés, et définir des alarmes qui vous informent ou prennent des mesures lorsqu'une métrique spécifique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation de certains APIs ou vous avertir si vous approchez de vos quotas de cryptographie des AWS paiements. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d' EC2 instances Amazon et d'autres sources. CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.
- AWS CloudTrailcapture les appels d'API et les événements associés effectués par ou pour le compte de votre AWS compte et envoie les fichiers journaux dans un compartiment Amazon S3 que vous spécifiez. Vous pouvez identifier les utilisateurs et les comptes appelés AWS, le point de terminaison appelé, les ressources (clés) utilisées, l'adresse IP source à partir de laquelle les appels ont été effectués et le moment où les appels ont eu lieu. Pour plus d'informations, consultez le AWS CloudTrail Guide de l'utilisateur .

Rubriques

- Enregistrement des appels de l'API AWS de cryptographie des paiements à l'aide de AWS CloudTrail

Enregistrement des appels de l'API AWS de cryptographie des paiements à l'aide de AWS CloudTrail

AWS La cryptographie des paiements est intégrée à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans le domaine de la cryptographie des AWS paiements. CloudTrail capture tous les appels d'API pour la cryptographie des AWS paiements sous forme d'événements. Les appels capturés incluent des appels de la console et les appels de code vers les opérations d'API . Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour le chiffrement des AWS paiements. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les derniers événements de gestion (plan de contrôle) dans la CloudTrail console dans l'historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été faite à AWS Payment Cryptography, l'adresse IP à partir de laquelle la demande a été faite, qui a fait la demande, quand elle a été faite et des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Rubriques

- AWS Informations relatives à la cryptographie des paiements dans CloudTrail
- Événements du plan de contrôle dans CloudTrail
- Événements liés aux données dans CloudTrail
- Comprendre les entrées AWS du fichier journal du plan de contrôle de la cryptographie des paiements
- Comprendre les entrées AWS du fichier journal du plan de données relatif à la cryptographie des paiements

AWS Informations relatives à la cryptographie des paiements dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans la cryptographie des AWS paiements, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez afficher, rechercher et télécharger les événements récents dans votre compte AWS . Pour

plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris les événements liés à la cryptographie des AWS paiements, créez une trace. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal de suivi dans la console, il s'applique à toutes les régions AWS . Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour plus d'informations, consultez les ressources suivantes :

- Présentation de la création d'un journal de suivi
- CloudTrail services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux provenant de plusieurs régions
- Réception de fichiers CloudTrail journaux provenant de plusieurs comptes

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été faite avec les informations d'identification de l'utilisateur root ou Gestion des identités et des accès AWS (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour de plus amples informations, veuillez consulter l'élément userIdentity CloudTrail .

Événements du plan de contrôle dans CloudTrail

CloudTrail enregistre les opérations AWS de cryptographie des paiements, telles que CreateKey, ImportKey,, DeleteKey, ListKeysTagResource, et toutes les autres opérations du plan de contrôle.

Événements liés aux données dans CloudTrail

Les événements de données fournissent des informations sur les opérations de ressources effectuées sur ou dans une ressource, telles que le chiffrement d'une charge utile ou la traduction d'un code PIN. Les événements de données sont des activités volumineuses qui CloudTrail ne sont pas enregistrées par défaut. Vous pouvez activer la journalisation des actions de l'API des événements de données pour les événements du plan de données AWS Payment Cryptography à l'aide de CloudTrail APIs notre console. Pour plus d'informations, consultez Journalisation des événements de données dans le Guide de l'utilisateur AWS CloudTrail .

Avec CloudTrail, vous devez utiliser des sélecteurs d'événements avancés pour décider quelles activités de l'API AWS de cryptographie des paiements sont enregistrées et enregistrées. Pour enregistrer les événements du plan de données de cryptographie des AWS paiements, vous devez inclure le type de ressource `AWS Payment Cryptography key` et`AWS Payment Cryptography alias`. Une fois cette configuration effectuée, vous pouvez peaufiner vos préférences de journalisation en spécifiant les événements de données à enregistrer, par exemple en utilisant le filtre `eventName` pour suivre les événements `EncryptData`. Pour plus d'informations, consultez `AdvancedEventSelector` dans la Référence d'API AWS CloudTrail .

> **Note**
>
> Pour vous abonner aux événements relatifs aux données de cryptographie des AWS paiements, vous devez utiliser des sélecteurs d'événements avancés. Nous vous recommandons de vous abonner aux événements clés et aux alias pour être sûr de recevoir tous les événements.

AWS Événements relatifs aux données de cryptographie des paiements :

- `DecryptData`
- `EncryptData`
- `GenerateCardValidationData`
- `GenerateMac`
- `GeneratePinData`
- `ReEncryptData`
- `TranslatePinData`
- `VerifyAuthRequestCryptogram`

- VerifyCardValidationData
- VerifyMac
- VerifyPinData

Des frais supplémentaires sont facturés pour les événements de données. Pour plus d'informations, consultez Tarification d'AWS CloudTrail .

Comprendre les entrées AWS du fichier journal du plan de contrôle de la cryptographie des paiements

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal illustrant l'`CreateKey`action AWS de chiffrement des paiements.

```
      {
    CloudTrailEvent: {
      tlsDetails= {
        TlsDetails: {
          cipherSuite=TLS_AES_128_GCM_SHA256,
          tlsVersion=TLSv1.3,
          clientProvidedHostHeader=controlplane.paymentcryptography.us-
west-2.amazonaws.com
        }
      },
    requestParameters=CreateKeyInput (
      keyAttributes=KeyAttributes(
        KeyUsage=TR31_B0_BASE_DERIVATION_KEY,
        keyClass=SYMMETRIC_KEY,
        keyAlgorithm=AES_128,
        keyModesOfUse=KeyModesOfUse(
          encrypt=false,
          decrypt=false,
```

```
          wrap=false
          unwrap=false,
          generate=false,
          sign=false,
          verify=false,
          deriveKey=true,
          noRestrictions=false)
        ),
      keyCheckValueAlgorithm=null,
      exportable=true,
      enabled=true,
      tags=null),
    eventName=CreateKey,
    userAgent=Coral/Apache-HttpClient5,
    responseElements=CreateKeyOutput(
      key=Key(
        keyArn=arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp,
          keyAttributes=KeyAttributes(
            KeyUsage=TR31_B0_BASE_DERIVATION_KEY,
            keyClass=SYMMETRIC_KEY,
            keyAlgorithm=AES_128,
            keyModesOfUse=KeyModesOfUse(
              encrypt=false,
              decrypt=false,
              wrap=false,
              unwrap=false,
              generate=false,
              sign=false,
              verify=false,
              deriveKey=true,
              noRestrictions=false)
            ),
          keyCheckValue=FE23D3,
          keyCheckValueAlgorithm=ANSI_X9_24,
          enabled=true,
          exportable=true,
          keyState=CREATE_COMPLETE,
          keyOrigin=AWS_PAYMENT_CRYPTOGRAPHY,
          createTimestamp=Sun May 21 18:58:32 UTC 2023,
          usageStartTimestamp=Sun May 21 18:58:32 UTC 2023,
          usageStopTimestamp=null,
          deletePendingTimestamp=null,
          deleteTimestamp=null)
```

```
        ),
      sourceIPAddress=192.158.1.38,
        userIdentity={
          UserIdentity: {
            arn=arn:aws:sts::111122223333:assumed-role/TestAssumeRole-us-west-2/
ControlPlane-IntegTest-68211a2a-3e9d-42b7-86ac-c682520e0410,
            invokedBy=null,
            accessKeyId=TESTXECZ5U2ZULLHHMJG,
            type=AssumedRole,
            sessionContext={
              SessionContext: {
                sessionIssuer={
                  SessionIssuer: {arn=arn:aws:iam::111122223333:role/TestAssumeRole-us-
west-2,
                  type=Role,
                  accountId=111122223333,
                  userName=TestAssumeRole-us-west-2,
                  principalId=TESTXECZ5U9M4LGF2N6Y5}
                },
                attributes={
                  SessionContextAttributes: {
                    creationDate=Sun May 21 18:58:31 UTC 2023,
                    mfaAuthenticated=false
                  }
                },
                webIdFederationData=null
              }
            },
          username=null,
          principalId=TESTXECZ5U9M4LGF2N6Y5:ControlPlane-User,
          accountId=111122223333,
          identityProvider=null
        }
      },
      eventTime=Sun May 21 18:58:32 UTC 2023,
      managementEvent=true,
      recipientAccountId=111122223333,
      awsRegion=us-west-2,
      requestID=151cdd67-4321-1234-9999-dce10d45c92e,
      eventVersion=1.08, eventType=AwsApiCall,
      readOnly=false,
      eventID=c69e3101-eac2-1b4d-b942-019919ad2faf,
      eventSource=payment-cryptography.amazonaws.com,
      eventCategory=Management,
```

```
      additionalEventData={
    }
  }
}
```

L'exemple suivant montre une entrée de CloudTrail journal qui illustre la cryptographie des AWS paiements permettant la réplication de clés multirégionales.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "payment-cryptography.amazonaws.com"
    },
    "eventTime": "2025-08-15T17:50:41Z",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventName": "SynchronizeMultiRegionKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "payment-cryptography.amazonaws.com",
    "userAgent": "payment-cryptography.amazonaws.com",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "55c0fcbc-5b2e-4bd2-a976-99305be6e6fc",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "keyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/key-id",
        "replicationRegion": "us-east-2"
    },
    "eventCategory": "Management"
}
```

Comprendre les entrées AWS du fichier journal du plan de données relatif à la cryptographie des paiements

Les événements du plan de données peuvent éventuellement être configurés et fonctionner de la même manière que les journaux du plan de contrôle, mais il s'agit généralement de volumes beaucoup plus importants. Compte tenu de la nature sensible de certaines entrées et sorties relatives

aux opérations du plan de données de cryptographie des AWS paiements, certains champs peuvent contenir le message « *** Données sensibles expurgées*** ». Ceci n'est pas configurable et vise à empêcher l'apparition de données sensibles dans les journaux ou les traces.

L'exemple suivant montre une entrée de CloudTrail journal illustrant l'`EncryptData`action AWS de chiffrement des paiements.

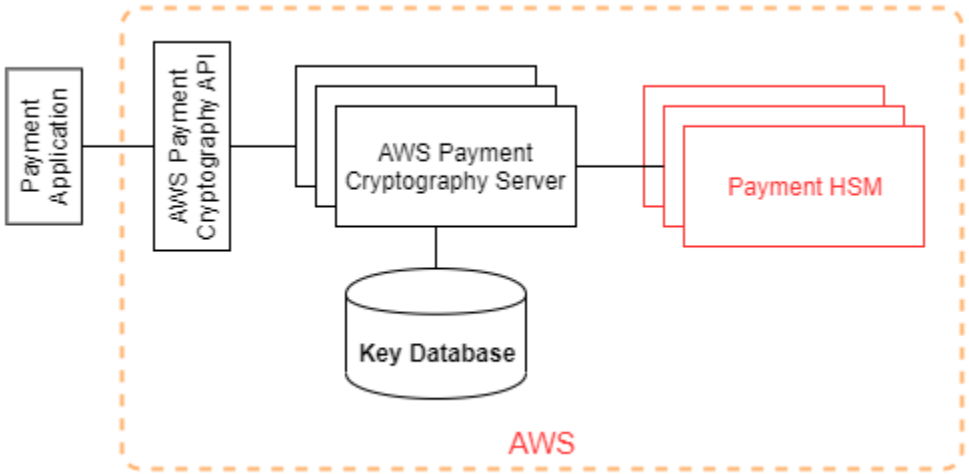

```
      {
        "Records": [
            {
                "eventVersion": "1.09",
                "userIdentity": {
                    "type": "AssumedRole",
                    "principalId": "TESTXECZ5U2ZULLHHMJG:DataPlane-User",
                    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/DataPlane-
User",
                    "accountId": "111122223333",
                    "accessKeyId": "TESTXECZ5U2ZULLHHMJG",
                    "userName": "",
                    "sessionContext": {
                        "sessionIssuer": {
                            "type": "Role",
                            "principalId": "TESTXECZ5U9M4LGF2N6Y5",
                            "arn": "arn:aws:iam::111122223333:role/Admin",
                            "accountId": "111122223333",
                            "userName": "Admin"
                        },
                        "attributes": {
                            "creationDate": "2024-07-09T14:23:05Z",
                            "mfaAuthenticated": "false"
                        }
                    }
                },
                "eventTime": "2024-07-09T14:24:02Z",
                "eventSource": "payment-cryptography.amazonaws.com",
                "eventName": "GenerateCardValidationData",
                "awsRegion": "us-east-2",
                "sourceIPAddress": "192.158.1.38",
                "userAgent": "aws-cli/2.17.6 md/awscrt#0.20.11 ua/2.0 os/macos#23.4.0
 md/arch#x86_64 lang/python#3.11.8 md/pyimpl#CPython cfg/retry-mode#standard md/
installer#exe md/prompt#off md/command#payment-cryptography-data.generate-card-
validation-data",
```

```
                "requestParameters": {
                    "key_identifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp",
                    "primary_account_number": "*** Sensitive Data Redacted ***",
                    "generation_attributes": {
                        "CardVerificationValue2": {
                            "card_expiry_date": "*** Sensitive Data Redacted ***"
                        }
                    }
                },
                "responseElements": null,
                "requestID": "f2a99da8-91e2-47a9-b9d2-1706e733991e",
                "eventID": "e4eb3785-ac6a-4589-97a1-babdd3d4dd95",
                "readOnly": true,
                "resources": [
                    {
                        "accountId": "111122223333",
                        "type": "AWS::PaymentCryptography::Key",
                        "ARN": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp"
                    }
                ],
                "eventType": "AwsApiCall",
                "managementEvent": false,
                "recipientAccountId": "111122223333",
                "eventCategory": "Data",
                "tlsDetails": {
                    "tlsVersion": "TLSv1.3",
                    "cipherSuite": "TLS_AES_128_GCM_SHA256",
                    "clientProvidedHostHeader": "dataplane.payment-cryptography.us-
east-2.amazonaws.com"
                }
            }
        ]
    }
```

Détails cryptographiques

AWS Payment Cryptography fournit une interface Web permettant de générer et de gérer des clés cryptographiques pour les transactions de paiement. AWS Payment Cryptography propose des services standard de gestion des clés et de cryptographie des transactions de paiement, ainsi que des outils que vous pouvez utiliser pour une gestion et un audit centralisés. Cette documentation fournit une description détaillée des opérations cryptographiques que vous pouvez utiliser dans AWS Payment Cryptography pour vous aider à évaluer les fonctionnalités proposées par le service.

AWS La cryptographie des paiements contient plusieurs interfaces (y compris une RESTful API, via la CLI AWS, le SDK AWS et le AWS Management Console) pour demander des opérations cryptographiques à un parc distribué de modules de sécurité matériels certifiés PCI PTS HSM.



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

AWS Les serveurs de cryptographie des paiements se connectent au HSM sous-jacent via un réseau privé non virtuel. Les connexions entre les composants du service et le HSM sont sécurisées par le protocole TLS mutuel (MTL) pour l'authentification et le chiffrement.

Rubriques

Objectifs de conception

AWS La cryptographie des paiements est conçue pour répondre aux exigences suivantes :

- Fiable — 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 de cryptographie AWS de paiement en texte clair. 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 Aucun employé d'Amazon n'a accès aux clés principales (ou principales) ou aux sauvegardes du HSM. Les clés principales ne peuvent pas être synchronisées avec celles HSMs qui ne font pas partie d'une région de cryptographie des AWS paiements. Toutes les autres clés sont protégées par des clés principales HSM. Par conséquent, les clés AWS de chiffrement des paiements du client ne sont pas utilisables en dehors du service de cryptographie des AWS paiements opérant sur le compte du client.
- Faible latence et haut débit — La cryptographie des AWS paiements fournit des opérations cryptographiques à des niveaux de latence et de débit adaptés à la gestion des clés cryptographiques de paiement et au traitement des transactions de paiement.
- Durabilité — La durabilité des clés cryptographiques est conçue pour être égale à celle des services les plus durables d'AWS. Une clé cryptographique unique peut être partagée avec un terminal de paiement, une carte à puce EMV ou un autre dispositif cryptographique sécurisé (SCD) utilisé depuis de nombreuses années.
- Régions indépendantes : AWS propose des régions indépendantes aux clients qui ont besoin de restreindre l'accès aux données dans différentes régions ou de se conformer aux exigences en matière de résidence des données. L'utilisation des clés peut être isolée au sein d'une région AWS.
- Source sécurisée de nombres aléatoires — Comme une cryptographie efficace dépend d'une génération de nombres aléatoires vraiment imprévisible, la cryptographie des AWS paiements fournit une source validée de nombres aléatoires de haute qualité. Toute génération de clés pour

la cryptographie des AWS paiements utilise un HSM certifié PCI PTS HSM, fonctionnant en mode PCI.

- Audit — La cryptographie des AWS paiements enregistre l'utilisation et la gestion des clés cryptographiques dans les CloudTrail journaux et les journaux de service disponibles via Amazon. CloudWatch Vous pouvez utiliser CloudTrail les journaux pour contrôler l'utilisation de vos clés cryptographiques, y compris l'utilisation des clés par les comptes avec lesquels vous avez partagé des clés. AWS La cryptographie des paiements est auditée par des évaluateurs tiers par rapport aux normes PCI applicables, aux marques de cartes et aux normes de sécurité des paiements régionales. Les attestations et les guides de responsabilité partagée sont disponibles sur AWS Artifact.
- Elastic — La cryptographie des AWS paiements évolue en fonction de votre demande. Au lieu de prévoir et de réserver la capacité HSM, AWS Payment Cryptography fournit une cryptographie de paiement à la demande. AWS Payment Cryptography est responsable du maintien de la sécurité et de la conformité du HSM afin de fournir une capacité suffisante pour répondre à la demande de pointe des clients.

Fondations

Les rubriques de ce chapitre décrivent les primitives cryptographiques de la cryptographie des AWS paiements et les endroits où elles sont utilisées. Ils présentent également les éléments de base du service.

Rubriques

Primitives cryptographiques

AWS La cryptographie des paiements utilise des algorithmes cryptographiques standard paramétrables afin que les applications puissent implémenter les algorithmes nécessaires à leur cas d'utilisation. L'ensemble des algorithmes cryptographiques est défini par les normes PCI, ANSI X9 et ISO EMVco. Toute la cryptographie est effectuée par la norme PCI PTS HSM répertoriée et exécutée en mode HSMs PCI.

Entropie et génération de nombres aléatoires

AWS La génération de la clé de cryptographie de paiement est effectuée sur la HSMs cryptographie AWS de paiement. Ils HSMs implémentent un générateur de nombres aléatoires qui répond aux exigences de la norme PCI PTS HSM pour tous les types de clés et paramètres pris en charge.

Opérations clés symétriques

Les algorithmes clés symétriques et les points forts définis dans les normes ANSI X9 TR 31, ANSI X9.24 et PCI PIN Annex C sont pris en charge :

- Fonctions de hachage — Algorithmes de la SHA3 famille SHA2 and dont la taille de sortie est supérieure à 2551. À l'exception de la rétrocompatibilité avec les terminaux pré-PCI PTS POI v3.
- Chiffrement et déchiffrement : AES avec une taille de clé supérieure ou égale à 128 bits, ou TDEA avec une taille de clé supérieure ou égale à 112 bits (2 clés ou 3 clés).
- Codes d'authentification de message (MACs) CMAC ou GMAC avec AES, ainsi que HMAC avec une fonction de hachage approuvée et une taille de clé supérieure ou égale à 128.

AWS La cryptographie des paiements utilise la norme AES 256 pour les clés principales HSM, les clés de protection des données et les clés de session TLS.

Remarque : Certaines des fonctions répertoriées sont utilisées en interne pour prendre en charge les protocoles et les structures de données standard. Consultez la documentation de l'API pour connaître les algorithmes pris en charge par des actions spécifiques.

Opérations clés asymétriques

Les algorithmes clés asymétriques et les points forts définis dans les normes ANSI X9 TR 31, ANSI X9.24 et PCI PIN Annex C sont pris en charge :

- Schémas d'établissement clés approuvés, tels que décrits dans le NIST SP8 00-56A (). ECC/ FCC2-based key agreement), NIST SP800-56B (IFC-based key agreement), and NIST SP800-38F (AES-based key encryption/wrapping

AWS Les hôtes de Payment Cryptography autorisent uniquement les connexions au service via le protocole TLS avec une suite de chiffrement garantissant une parfaite confidentialité des transmissions.

Remarque : Certaines des fonctions répertoriées sont utilisées en interne pour prendre en charge les protocoles et les structures de données standard. Consultez la documentation de l'API pour connaître les algorithmes pris en charge par des actions spécifiques.

Rangement des clés

AWS Les clés de cryptographie de paiement sont protégées par des clés principales HSM AES 256 et stockées dans des blocs de clés ANSI X9 TR 31 dans une base de données cryptée. La base de données est répliquée dans une base de données en mémoire sur les serveurs de cryptographie des AWS paiements.

Selon l'annexe C de la norme de sécurité PCI PIN, les clés AES 256 sont aussi puissantes ou plus puissantes que :

- TDEA à 3 touches
- RSA 15360 bits
- ECC 512 bits
- DSA, DH et MQV 15360/512

Importation de clés à l'aide de clés symétriques

AWS La cryptographie des paiements prend en charge l'importation de cryptogrammes et de blocs de clés dotés de clés symétriques ou publiques avec une clé de chiffrement à clé symétrique (KEK) aussi forte ou plus puissante que la clé protégée à importer.

Importation de clés à l'aide de clés asymétriques

AWS La cryptographie des paiements prend en charge l'importation de cryptogrammes et de blocs de clés dotés de clés symétriques ou publiques protégées par une clé de chiffrement à clé privée

(KEK) aussi forte ou plus puissante que la clé protégée à importer. L'authenticité et l'intégrité de la clé publique fournie pour le déchiffrement doivent être garanties par un certificat délivré par une autorité de confiance du client.

Les KEK publics fournis par AWS Payment Cryptography disposent de l'authentification et de la protection d'intégrité d'une autorité de certification (CA) avec une conformité attestée à la sécurité par code PIN PCI et à l'annexe A de la norme PCI P2PE.

Exportation de clés

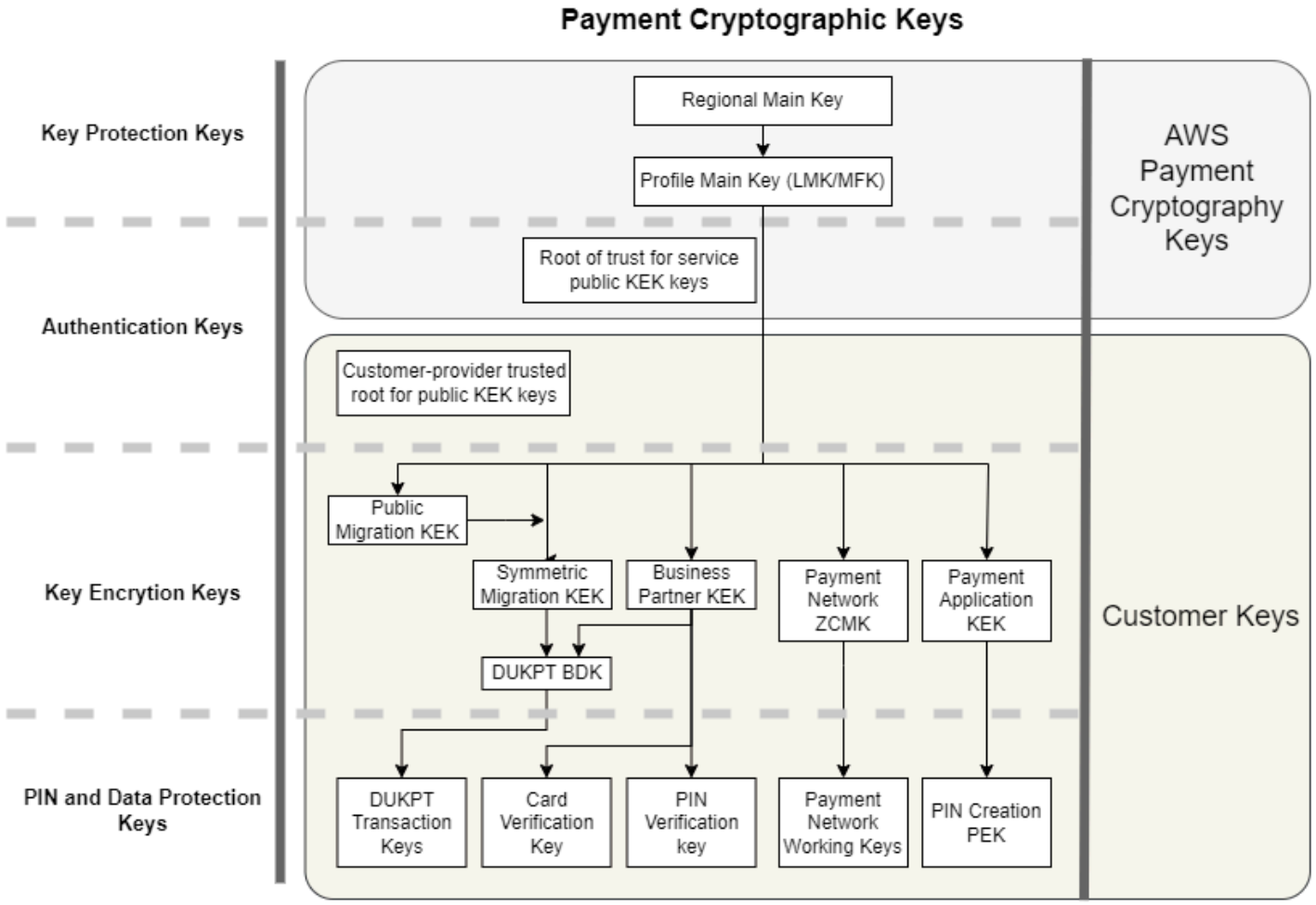
Les clés peuvent être exportées et protégées par des clés dotées de la valeur appropriée KeyUsage et qui sont aussi fortes ou plus fortes que la clé à exporter.

Protocole DUKPT (clé unique dérivée par transaction)

AWS La cryptographie des paiements prend en charge les clés de dérivation de base (BDK) TDEA et AES, comme décrit par la norme ANSI X9.24-3.

Hiérarchie des clés

La hiérarchie des clés de chiffrement des AWS paiements garantit que les clés sont toujours protégées par des clés aussi fortes ou plus fortes que les clés qu'elles protègent.



AWS Les clés de cryptographie de paiement sont utilisées pour la protection des clés au sein du service :

Clé	Description
Clé principale régionale	Protège les images ou profils HSM virtuels utilisés pour le traitement cryptographique. Cette clé n'existe que dans le HSM et les sauvegardes sécurisées.
Clé principale du profil	Clé de protection de clé client de haut niveau, traditionnellement appelée clé principale locale (LMK) ou clé de fichier principale (MFK) pour les clés client. Cette clé n'existe que dans le HSM et les sauvegardes sécurisées. Les profils

Clé	Description
	définissent des configurations HSM distinctes conformément aux normes de sécurité pour les cas d'utilisation des paiements.
Racine de confiance pour les clés de chiffrement à clé publique (KEK) de cryptographie des AWS paiements	La clé publique racine fiable et le certificat d'authentification et de validation des clés publiques fournis par AWS Payment Cryptography pour l'importation et l'exportation de clés à l'aide de clés asymétriques.

Les clés client sont regroupées en fonction des clés utilisées pour protéger les autres clés et des clés qui protègent les données relatives au paiement. Voici des exemples de clés client des deux types :

Clé	Description
Root sécurisé fourni par le client pour les clés KEK publiques	Clé publique et certificat fournis par vous en tant que base de confiance pour authentifier et valider les clés publiques que vous fournissez pour l'importation et l'exportation de clés à l'aide de clés asymétriques.
Clés de chiffrement clés (KEK)	Les KEK sont utilisés uniquement pour chiffrer d'autres clés destinées à être échangées entre des magasins de clés externes et AWS Payment Cryptography, des partenaires commerciaux, des réseaux de paiement ou différentes applications au sein de votre organisation.
Clé unique dérivée par transaction (DUKPT) Clé de dérivation de base (BDK)	BDKs sont utilisés pour créer des clés uniques pour chaque terminal de paiement et traduire les transactions provenant de plusieurs terminaux en une seule clé fonctionnelle de la banque acquéreuse, ou de l'acquéreur. La meilleure pratique, requise par le Point-to-Point

Clé	Description
	chiffrement PCI (P2PE), est d'utiliser différents BDKs modèles de terminaux, services d'injecti on de clés ou d'initialisation, ou toute autre segmentation afin de limiter l'impact de la compromission d'un BDK.
Clé principale de contrôle de zone du réseau de paiement (ZCMK)	Les ZCMK, également appelés clés de zone ou clés principales de zone, sont fournis par les réseaux de paiement pour établir les clés de travail initiales.
Clés de transaction DUKPT	Les terminaux de paiement configurés pour DUKPT obtiennent une clé unique pour le terminal et la transaction. Le HSM recevant la transaction peut déterminer la clé à partir de l'identifiant du terminal et du numéro de séquence de transaction.
Clés de préparation des données des cartes	Les clés principales de l'émetteur EMV, les clés de carte EMV et les valeurs de vérificat ion, ainsi que les clés de protection des fichiers de données de personnalisation des cartes sont utilisées pour créer des données pour des cartes individuelles destinées à être utilisées par un fournisseur de personnalisation de cartes. Ces clés et données de validation cryptographique sont également utilisées par les banques émettrices, ou émetteurs, pour authentifier les données des cartes dans le cadre de l'autorisation des transactions.

Clé	Description
Clés de préparation des données des cartes	Les clés principales de l'émetteur EMV, les clés de carte EMV et les valeurs de vérificat ion, ainsi que les clés de protection des fichiers de données de personnalisation des cartes sont utilisées pour créer des données pour des cartes individuelles destinées à être utilisées par un fournisseur de personnalisation de cartes. Ces clés et données de validation cryptographique sont également utilisées par les banques émettrices, ou émetteurs, pour authentifier les données des cartes dans le cadre de l'autorisation des transactions.
Clés de fonctionnement du réseau de paiement	Souvent appelées clé de travail de l'émetteu r ou clé de travail de l'acquéreur, ces clés cryptent les transactions envoyées ou reçues depuis les réseaux de paiement. Ces clés font l'objet d'une rotation fréquente par le réseau, souvent tous les jours ou toutes les heures. Il s'agit de clés de chiffrement par code PIN (PEK) pour les PIN/Debit transactions.
Clés de chiffrement (PEK) par numéro d'identif ication personnel (PIN)	Les applications qui créent ou déchiffrent des blocs de code PIN utilisent le PEK pour empêcher le stockage ou la transmission de code PIN en texte clair.

Opérations internes

Cette rubrique décrit les exigences internes mises en œuvre par le service pour sécuriser les clés des clients et les opérations cryptographiques dans le cadre d'un service de cryptographie et de gestion des clés de paiement évolutif et distribué à l'échelle mondiale.

Rubriques

Protection du HSM

Spécifications et cycle de vie du HSM

AWS La cryptographie des paiements utilise une flotte de solutions disponibles dans le commerce. HSMs Ils HSMs sont validés par la norme FIPS 140-2 niveau 3 et utilisent également des versions de microprogramme et la politique de sécurité répertoriée sur la liste des périphériques PCI PTS approuvée par le PCI Security Standards Council comme étant conformes à la norme PCI HSM v3. La norme PCI PTS HSM inclut des exigences supplémentaires pour la fabrication, l'expédition, le déploiement, la gestion et la destruction du matériel HSM, qui sont importantes pour la sécurité et la conformité des paiements, mais qui ne sont pas traitées par la norme FIPS 140.

Des évaluateurs tiers vérifient le modèle du HSM, le microprogramme, la configuration, la gestion physique du cycle de vie, le contrôle des modifications, les contrôles d'accès des opérateurs, la gestion des clés principales et toutes les exigences PCI PIN et P2PE liées aux HSMs opérations HSM.

Tous fonctionnent en mode PCI et HSMs sont configurés selon la politique de sécurité PCI PTS HSM. Seules les fonctions nécessaires pour prendre en charge les cas d'utilisation de la cryptographie des AWS paiements sont activées. AWS La cryptographie des paiements ne permet pas d'imprimer, d'afficher ou de renvoyer du texte PINs clair.

Sécurité physique des appareils HSM

Le service ne peut utiliser HSMs que les clés de l'appareil signées par une autorité de certification de chiffrement des AWS paiements (CA) du fabricant avant la livraison. La cryptographie des AWS paiements est une sous-autorité de certification de l'autorité de certification du fabricant qui est à l'origine de la confiance en ce qui concerne les certificats des fabricants et des appareils HSM. L'autorité de certification du fabricant a attesté la conformité à l'annexe A de sécurité du code PIN PCI et à l'annexe A de la norme PCI P2PE. Le fabricant vérifie que tous les HSM dotés de clés

d'appareil signées par l'autorité de certification de chiffrement des AWS paiements sont expédiés au destinataire désigné par AWS.

Conformément à la norme PCI PIN Security, le fabricant fournit une liste de numéros de série via un canal de communication différent de celui utilisé pour l'expédition du HSM. Ces numéros de série sont vérifiés à chaque étape du processus d'installation du HSM dans les centres de données AWS. Enfin, les opérateurs de chiffrement des AWS paiements valident la liste des HSM installés par rapport à la liste du fabricant avant d'ajouter le numéro de série à la liste des HSM autorisés à recevoir des clés de chiffrement des AWS paiements.

HSMs sont stockés de manière sécurisée ou font l'objet d'un double contrôle à tout moment, ce qui inclut :

- Expédition par le fabricant vers une installation d'assemblage de racks AWS.
- Pendant le montage du rack.
- Expédition depuis l'installation d'assemblage des racks vers un centre de données.
- Réception et installation dans la salle de traitement sécurisée d'un centre de données. Les racks HSM assurent un double contrôle avec des serrures à accès contrôlé par carte, des capteurs de porte avec alarme et des caméras.
- Pendant les opérations.
- Pendant la mise hors service et la destruction.

Un système complet chain-of-custody, avec une responsabilité individuelle, est maintenu et contrôlé pour chaque HSM.

Initialisation du HSM

Un HSM n'est initialisé dans le cadre de la flotte de cryptographie des AWS paiements qu'une fois que son identité et son intégrité ont été validées par des numéros de série, des clés de périphérique installées par le fabricant et une somme de contrôle du microprogramme. Une fois l'authenticité et l'intégrité d'un HSM validées, celui-ci est configuré, notamment en activant le mode PCI. Ensuite, les clés principales de la région de cryptographie des AWS paiements et les clés principales du profil sont établies et le HSM est mis à la disposition du service.

Service et réparation HSM

Les HSM comportent des composants réparables qui ne nécessitent pas de violation de la limite cryptographique du périphérique. Ces composants incluent les ventilateurs de refroidissement, les

blocs d'alimentation et les batteries. Si un HSM ou un autre périphérique du rack HSM a besoin d'être réparé, le double contrôle est maintenu pendant toute la période d'ouverture du rack.

Mise hors service du HSM

La mise hors service est due à une défaillance end-of-life ou à une défaillance d'un HSM. Les HSM sont logiquement mis à zéro avant d'être retirés de leur rack, s'ils sont fonctionnels, puis détruits dans les salles de traitement sécurisées des centres de données AWS. Ils ne sont jamais renvoyés au fabricant pour réparation, utilisés à d'autres fins ou retirés d'une salle de traitement sécurisée avant d'être détruits.

Mise à jour du microprogramme HSM

Les mises à jour du microprogramme HSM sont appliquées lorsque cela est nécessaire pour maintenir l'alignement avec les versions répertoriées PCI PTS HSM et FIPS 140-2 (ou FIPS 140-3), si une mise à jour est liée à la sécurité ou s'il est déterminé que les clients peuvent bénéficier des fonctionnalités d'une nouvelle version. AWS Payment Cryptography HSMs exécute le off-the-shelf microprogramme correspondant aux versions répertoriées par PCI PTS HSM. L'intégrité des nouvelles versions du microprogramme est validée avec les versions de microprogramme certifiées PCI ou FIPS, puis leur fonctionnalité est testée avant d'être déployées auprès de tous. HSMs

Accès de l'opérateur

Les opérateurs peuvent accéder au HSM sans console à des fins de dépannage dans de rares cas où les informations recueillies auprès du HSM au cours des opérations normales ne sont pas suffisantes pour identifier un problème ou planifier une modification. Les étapes suivantes sont exécutées :

- Les activités de dépannage sont développées et approuvées et la session hors console est planifiée.
- Un HSM est supprimé du service de traitement des clients.
- Les touches principales sont supprimées, sous double contrôle.
- L'opérateur est autorisé à accéder au HSM sans console pour effectuer des activités de dépannage approuvées, sous double contrôle.
  - Après la fin de la session hors console, le processus de provisionnement initial est effectué sur le HSM, en renvoyant le microprogramme et la configuration standard, puis en synchronisant la clé principale, avant de renvoyer le HSM au service des clients.

- Les enregistrements de la session sont enregistrés dans le suivi des modifications.
- Les informations obtenues lors de la session sont utilisées pour planifier les modifications futures.

Tous les enregistrements d'accès hors console sont examinés pour vérifier la conformité des processus et les modifications potentielles apportées à la surveillance du HSM, au processus de non-console-access gestion ou à la formation des opérateurs.

Gestion générale des clés

Tous les HSM d'une région sont synchronisés avec une clé principale de région. Une clé principale de région protège au moins une clé principale de profil. Une clé principale de profil protège les clés des clients.

Toutes les clés principales sont générées par un HSM et distribuées par distribution de clés symétrique à l'aide de techniques asymétriques, conformément à la norme ANSI X9 TR 34 et à l'annexe A du code PIN PCI.

Génération

Les clés principales AES 256 bits sont générées sur l'un des HSM fournis pour le parc de services HSM, à l'aide du générateur de nombres aléatoires PCI PTS HSM.

Synchronisation des touches principales de la région

Les clés principales de la région HSM sont synchronisées par le service sur l'ensemble de la flotte régionale avec les mécanismes définis par la norme ANSI X9 TR-34, notamment :

- Authentification mutuelle à l'aide de clés et de certificats de l'hôte de distribution de clés (KDH) et du dispositif de réception de clés (KRD) pour garantir l'authentification et l'intégrité des clés publiques.
- Les certificats sont signés par une autorité de certification (CA) qui répond aux exigences de l'annexe A2 du code PIN PCI, à l'exception des algorithmes asymétriques et des points forts de clé appropriés pour protéger les clés AES 256 bits.
- L'identification et la protection des clés symétriques distribuées sont conformes à la norme ANSI X9 TR-34 et à l'annexe A1 du code PIN PCI, à l'exception des algorithmes asymétriques et des atouts de clé appropriés pour protéger les clés AES 256 bits.

Les clés principales de région sont établies pour HSMs celles qui ont été authentifiées et fournies pour une région par :

- Une clé principale est générée sur un HSM de la région. Ce HSM est désigné comme hôte de distribution des clés.
- Tous les éléments fournis HSMs dans la région génèrent un jeton d'authentification KRD, qui contient la clé publique du HSM et des informations d'authentification non rejouables.
- Les jetons KRD sont ajoutés à la liste d'autorisation du KDH une fois que le KDH a validé l'identité et l'autorisation du HSM de recevoir des clés.
- Le KDH produit un jeton de clé principale authentifiable pour chaque HSM. Les jetons contiennent des informations d'authentification KDH et une clé principale cryptée qui ne peut être chargée que sur un HSM pour lequel elle a été créée.
- Chaque HSM reçoit le jeton de clé principal conçu pour lui. Après avoir validé les propres informations d'authentification du HSM et les informations d'authentification KDH, la clé principale est déchiffrée par la clé privée KRD et chargée dans la clé principale.

Dans le cas où un seul HSM doit être resynchronisé avec une région :

- Il est revalidé et approvisionné avec le microprogramme et la configuration.
- S'il s'agit d'un nouveau produit dans la région :
  - Le HSM génère un jeton d'authentification KRD.
  - Le KDH ajoute le jeton à sa liste d'autorisations.
  - Le KDH génère un jeton de clé principal pour le HSM.
  - Le HSM charge la clé principale.
  - Le HSM est mis à la disposition du service.

Cela garantit que :

- Seul le HSM validé pour le traitement AWS de cryptographie des paiements dans une région peut recevoir la clé principale de cette région.
- Seule une clé principale provenant d'un HSM de chiffrement des AWS paiements peut être distribuée à un HSM de la flotte.

Rotation des clés principales de la région

Les clés principales des régions font l'objet d'une rotation à l'expiration de la période de chiffrement, dans le cas peu probable d'une compromission présumée de la clé ou après des modifications du service susceptibles d'avoir un impact sur la sécurité de la clé.

Une nouvelle clé principale de région est générée et distribuée comme lors du provisionnement initial. Les clés principales du profil enregistrées doivent être converties en la clé principale de la nouvelle région.

La rotation des principales clés de la région n'a aucune incidence sur le traitement des clients.

Synchronisation des touches principales du profil

Les clés principales du profil sont protégées par les clés principales des régions. Cela limite un profil à une région spécifique.

Les clés principales du profil sont configurées en conséquence :

- Une clé principale de profil est générée sur un HSM dont la clé principale de région est synchronisée.
- La clé principale du profil est stockée et cryptée avec la configuration du profil et d'autres contextes.
- Le profil est utilisé pour les fonctions cryptographiques des clients par n'importe quel HSM de la région doté de la clé principale de région.

Rotation de la clé principale du profil

Les clés principales du profil font l'objet d'une rotation à l'expiration de la période de chiffrement, en cas de suspicion de compromission de la clé ou après des modifications du service susceptibles d'avoir un impact sur la sécurité de la clé.

Étapes de rotation :

- Une nouvelle clé principale de profil est générée et distribuée en tant que clé principale en attente, comme lors du provisionnement initial.
- Un processus d'arrière-plan traduit les informations clés du client de la clé principale du profil établie à la clé principale en attente.

- Lorsque toutes les clés du client ont été chiffrées avec la clé en attente, la clé en attente est promue clé principale du profil.
- Un processus en arrière-plan supprime le contenu clé du client protégé par la clé expirée.

La rotation des clés principales du profil n'a aucune incidence sur le traitement des clients.

Protection

Les clés ne dépendent que de la hiérarchie des clés pour la protection. La protection des clés principales est essentielle pour éviter de perdre ou de compromettre toutes les clés des clients.

Les clés principales de région peuvent être restaurées à partir d'une sauvegarde uniquement vers un HSM authentifié et provisionné pour le service. Ces clés ne peuvent être stockées que sous forme de jetons de clé principale chiffrés et authentifiables mutuellement provenant d'un KDH spécifique pour un HSM spécifique.

Les clés principales du profil sont stockées avec la configuration du profil et les informations contextuelles chiffrées par région.

Les clés client sont stockées dans des blocs de clés, protégés par une clé principale de profil.

Toutes les clés existent exclusivement dans un HSM ou sont stockées protégées par une autre clé de puissance cryptographique égale ou supérieure.

Durabilité

Les clés client pour la cryptographie des transactions et les fonctions commerciales doivent être disponibles même dans des situations extrêmes susceptibles de provoquer des pannes. AWS La cryptographie des paiements utilise un modèle de redondance à plusieurs niveaux dans les zones de disponibilité et les régions. AWS Les clients qui exigent une disponibilité et une durabilité des opérations cryptographiques de paiement supérieures à celles fournies par le service doivent mettre en œuvre des architectures multirégionales.

L'authentification HSM et les jetons de clé principale sont enregistrés et peuvent être utilisés pour restaurer une clé principale ou synchroniser avec une nouvelle clé principale, dans le cas où un HSM doit être réinitialisé. Les jetons sont archivés et utilisés uniquement sous double contrôle lorsque cela est nécessaire.

Accès de l'opérateur aux clés principales du HSM

Les clés principales n'existent que dans le HSM géré par le service et sécurisé dans des installations AWS sécurisées. Les clés principales ne peuvent pas être exportées depuis un HSM ou synchronisées avec un HSM qui n'est pas initialisé par le fabricant pour être utilisé dans le service. Les opérateurs AWS ne peuvent pas obtenir de clés principales sous quelque forme que ce soit qui pourrait être chargée dans un HSM non géré par le service.

Gestion des clés clients

Chez AWS, la confiance du client est notre priorité absolue. Vous gardez le contrôle total des clés que vous importez ou créez dans le service sous votre compte AWS. Vous êtes responsable de la configuration de l'accès aux clés.

AWS Payment Cryptography est un fournisseur de services qui utilise HSMs et gère les clés pour le compte des clients, à l'instar des fournisseurs de services de paiement de longue date. Le service est entièrement responsable de la sécurité physique et logique du HSM. La responsabilité de gestion des clés est partagée entre le service et les clients, car le client doit fournir des informations précises sur les clés créées ou importées par le service, que le service utilise pour garantir une utilisation et une gestion correctes des clés. Les protections de ségrégation des données AWS sont utilisées pour garantir que la compromission des clés appartenant à un compte AWS ne peut pas compromettre les clés appartenant à un autre.

AWS Payment Cryptography est entièrement responsable de la conformité physique du HSM et de la gestion des clés pour les clés gérées par le service. Cela nécessite la propriété et la gestion des clés principales du HSM et la protection des clés clients gérées par la cryptographie des AWS paiements.

Séparation de l'espace clé pour le client

AWS La cryptographie des paiements applique des politiques relatives aux clés pour toutes les utilisations des clés, notamment en limitant les principaux au compte propriétaire de la clé, sauf si une clé est explicitement partagée avec un autre compte.

Les comptes AWS assurent une séparation complète de l'environnement entre les clients ou les applications, de la même manière que les implémentations hors cloud dans différents centres de données. Chaque compte fournit un contrôle d'accès isolé, un réseau, des ressources informatiques, un stockage de données, des clés cryptographiques pour la protection des données et les transactions de paiement, ainsi que toutes les ressources AWS. Les services AWS tels que

Organizations et Control Tower permettent à l'entreprise de gérer des comptes d'applications distincts, comme des cages ou des salles au sein d'un centre de données d'entreprise.

Accès de l'opérateur aux clés du client

Les clés client gérées par le service sont stockées protégées par des clés principales de partition et ne peuvent être utilisées que par le compte client propriétaire ou le compte que le propriétaire a spécifiquement configuré pour le partage de clés. Les opérateurs AWS ne peuvent pas exporter ou effectuer de gestion de clés ou d'opérations cryptographiques avec les clés des clients en utilisant un accès manuel au service, qui est géré par les mécanismes d'accès manuel des opérateurs AWS.

Le code de service qui met en œuvre la gestion et l'utilisation des clés client est soumis aux pratiques du code de sécurité AWS, telles qu'évaluées dans le cadre de l'évaluation PCI DSS d'AWS.

Sauvegarde et restauration

Les clés et les informations clés stockées en interne par le service pour une région sont sauvegardées dans des archives cryptées par AWS. Les archives nécessitent un double contrôle par AWS pour être restaurées.

Blocs clés

Toutes les clés sont stockées et traitées dans des blocs de clés au format ANSI X9.143.

Les clés peuvent être importées dans le service à partir de cryptogrammes ou d'autres formats de blocs de clés pris en charge par ImportKey. De même, les clés peuvent être exportées, si elles sont exportables, vers d'autres formats de blocs de clés ou cryptogrammes pris en charge par les profils d'exportation de clés.

Utilisation des clés

L'utilisation des clés est limitée à celle configurée KeyUsage par le service. Le service échouera à toute demande impliquant une utilisation de clé, un mode d'utilisation ou un algorithme inappropriés pour l'opération cryptographique demandée.

Principales relations d'échange

La sécurité PCI PIN et la norme PCI P2PE exigent que les entreprises qui partagent des clés de chiffrement PINs ou des données de cartes, y compris les clés d'échange de clés (KEK) utilisées pour partager ces clés, ne partagent pas les mêmes clés avec d'autres organisations. Il est recommandé que les clés symétriques ne soient partagées qu'entre deux parties dans un seul but,

y compris au sein d'une même organisation. Cela permet de minimiser l'impact des compromissions présumées qui obligent à remplacer les clés concernées.

Même les analyses de rentabilisation qui nécessitent le partage de clés entre plus de deux parties devraient limiter le nombre de parties au minimum.

AWS La cryptographie des paiements fournit des balises clés qui peuvent être utilisées pour suivre et appliquer l'utilisation des clés conformément à ces exigences.

Par exemple, les clés KEK et BDK pour différentes installations d'injection de clés peuvent être identifiées en définissant un « KIF » = « POSStation » pour toutes les clés partagées avec ce fournisseur de services. Un autre exemple serait de marquer les clés partagées avec les réseaux de paiement avec « Network » = « PayCard ». Le balisage vous permet de créer des contrôles d'accès et de créer des rapports d'audit pour appliquer et démontrer vos principales pratiques de gestion.

Suppression de la clé

DeleteKey marque les clés de la base de données pour suppression après une période configurée par le client. Passé ce délai, la clé est irrémédiablement supprimée. Il s'agit d'un mécanisme de sécurité destiné à empêcher la suppression accidentelle ou malveillante d'une clé. Les touches marquées pour suppression ne sont disponibles que pour les actions RestoreKey.

Les clés supprimées restent dans les sauvegardes du service pendant 7 jours après leur suppression. Ils ne sont pas restaurables pendant cette période.

Les clés appartenant à des comptes AWS fermés sont marquées pour suppression. Si le compte est réactivé avant la fin de la période de suppression, toutes les clés marquées pour suppression sont restaurées, mais désactivées. Vous devez les réactiver pour pouvoir les utiliser pour des opérations cryptographiques.

Sécurité des communications

Externe

AWS Les points de terminaison de l'API de cryptographie des paiements répondent aux normes de AWS sécurité, notamment le protocole TLS 1.2 ou supérieur et la version 4 de Signature pour l'authentification et l'intégrité des demandes.

Les connexions TLS entrantes sont interrompues sur les équilibreurs de charge réseau et transmises aux gestionnaires d'API via des connexions TLS internes.

Internal (Interne)

Les communications internes entre les composants de service et entre les composants de service et les autres services AWS sont protégées par le protocole TLS à l'aide d'une cryptographie renforcée.

Les HSM se trouvent sur un réseau privé non virtuel accessible uniquement à partir de composants de service. Toutes les connexions entre le HSM et les composants de service sont sécurisées par le protocole TLS mutuel (MTL), égal ou supérieur au protocole TLS 1.2. Les certificats internes pour TLS et MTL sont gérés par Amazon Certificate Manager à l'aide d'une autorité de certification privée AWS. Le réseau interne VPCs et le réseau HSM sont surveillés pour détecter les activités inattendues et les modifications de configuration.

Journalisation et surveillance

Les journaux de service internes incluent :

- CloudTrail journaux des appels de service AWS effectués par le service
- CloudWatch journaux des deux événements directement enregistrés dans les CloudWatch journaux ou des événements depuis HSM
- Fichiers journaux du HSM et des systèmes de service
- Archives du journal

Toutes les sources de journaux surveillent et filtrent les informations sensibles, y compris celles relatives aux clés. Les journaux sont systématiquement examinés pour s'assurer qu'ils ne contiennent pas d'informations sensibles sur les clients.

L'accès aux journaux est limité aux personnes nécessaires pour remplir les rôles professionnels.

Tous les journaux sont conservés conformément aux politiques de conservation des journaux d'AWS.

Opérations auprès des clients

AWS Payment Cryptography est entièrement responsable de la conformité physique du HSM aux normes PCI. Le service fournit également un magasin de clés sécurisé et garantit que les clés ne peuvent être utilisées qu'aux fins autorisées par les normes PCI et spécifiées par vous lors de la création ou de l'importation. Vous êtes responsable de la configuration des attributs clés et de l'accès afin de tirer parti des fonctionnalités de sécurité et de conformité du service.

Rubriques

Génération de clés

Lorsque vous créez des clés, vous définissez les attributs que le service utilise pour garantir une utilisation conforme de la clé :

- Algorithme et longueur de clé
- Usage
- Disponibilité et expiration

Les balises utilisées pour le contrôle d'accès basé sur les attributs (ABAC) sont utilisées pour limiter les clés à utiliser avec des partenaires spécifiques ou les applications doivent également être définies lors de la création. Veillez à inclure des politiques limitant les rôles autorisés à supprimer ou à modifier des balises.

Vous devez vous assurer que les politiques qui déterminent les rôles autorisés à utiliser et à gérer la clé sont définies avant la création de la clé.

> **Note**
>
> Les politiques IAM relatives aux CreateKey commandes peuvent être utilisées pour appliquer et démontrer un double contrôle pour la génération de clés.

Importation de clés

Lors de l'importation de clés, les attributs destinés à garantir une utilisation conforme de la clé sont définis par le service à l'aide des informations cryptographiquement liées contenues dans le bloc de clés. Le mécanisme de définition du contexte clé fondamental consiste à utiliser des blocs clés créés

avec le HSM source et protégés par un KEK partagé ou asymétrique. Cela correspond aux exigences du code PIN PCI et préserve l'utilisation, l'algorithme et la force clé de l'application source.

Les attributs clés, les balises et les politiques de contrôle d'accès importants doivent être établis lors de l'importation, en plus des informations contenues dans le bloc clé.

L'importation de clés à l'aide de cryptogrammes ne transfère pas les attributs de clé depuis l'application source. Vous devez définir les attributs de manière appropriée en utilisant ce mécanisme.

Les clés sont souvent échangées à l'aide de composants en texte clair, transmises par les dépositaires des clés, puis chargées lors d'une cérémonie mettant en œuvre un double contrôle dans une pièce sécurisée. Cela n'est pas directement pris en charge par AWS Payment Cryptography. L'API exportera une clé publique avec un certificat qui peut être importé par votre propre HSM pour exporter un bloc de clés importable par le service. Vous permet d'utiliser votre propre HSM pour charger des composants en texte clair.

Vous devez utiliser les valeurs de contrôle des clés (KCV) pour vérifier que les clés importées correspondent aux clés source.

Les politiques IAM sur l' ImportKey API peuvent être utilisées pour appliquer et démontrer le double contrôle pour l'importation de clés.

Exportation de clés

Le partage de clés avec des partenaires ou des applications locales peut nécessiter l'exportation de clés. L'utilisation de blocs clés pour les exportations permet de maintenir le contexte clé fondamental avec le contenu clé crypté.

Les tags clés peuvent être utilisés pour limiter l'exportation vers KEK de clés partageant le même tag et la même valeur.

AWS La cryptographie des paiements ne fournit ni n'affiche les composants clés en texte clair. Cela nécessite un accès direct des dépositaires de clés aux périphériques cryptographiques sécurisés (SCD) certifiés PCI PTS HSM ou ISO 13491 pour l'affichage ou l'impression. Vous pouvez établir un KEK asymétrique ou un KEK symétrique avec votre SCD pour organiser la cérémonie de création des composants clés en texte clair sous double contrôle.

Les valeurs de contrôle des clés (KCV) doivent être utilisées pour vérifier que les clés source importées par le HSM de destination correspondent.

Suppression de clés

Vous pouvez utiliser l'API de suppression des clés pour planifier la suppression des clés après une période que vous avez configurée. Avant cette date, les clés sont récupérables. Une fois les clés supprimées, elles sont définitivement supprimées du service.

Les politiques IAM sur l' DeleteKey API peuvent être utilisées pour appliquer et démontrer un double contrôle pour la suppression des clés.

Rotating keys

L'effet de rotation des clés peut être implémenté à l'aide d'un alias de clé en créant ou en important une nouvelle clé, puis en modifiant l'alias de clé pour qu'il fasse référence à la nouvelle clé. L'ancienne clé serait supprimée ou désactivée, selon vos pratiques de gestion.

Quotas pour AWS Payment Cryptography

Votre compte AWS dispose de quotas par défaut, anciennement appelés limites, pour chaque service AWS. Sauf indication contraire, chaque quota est spécifique à une région. Vous pouvez demander des augmentations pour certains quotas, et d'autres quotas ne peuvent pas être augmentés.

Nom	Par défaut	Ajusta	Description
les alias ;	Chaque Région prise en charge : 2 000	Oui	Le nombre maximum d'alias que vous pouvez avoir sur ce compte dans la région actuelle.
Taux combiné de demandes de plan de contrôle	Chaque région prise en charge : 5 par seconde	Oui	Le nombre maximum de demandes de plan de contrôle par seconde que vous pouvez effectuer sur ce compte dans la région actuelle. Ce quota s'applique à toutes les opérations du plan de contrôle combinées.
Taux combiné de demandes de plan de données (asymétrique)	Chaque région prise en charge : 20 par seconde	Oui	Le nombre maximum de demandes par seconde pour les opérations du plan de données avec une clé asymétrique que vous pouvez effectuer dans ce compte dans la région actuelle. Ce quota s'applique à toutes les opérations du plan de données combinées.

Nom	Par défaut	Ajusta	Description
Taux combiné de demandes de plan de données (symétrique)	Chaque Région prise en charge : 500 par seconde	Oui	Le nombre maximum de demandes par seconde pour les opérations du plan de données avec une clé symétrique que vous pouvez effectuer dans ce compte dans la région actuelle. Ce quota s'applique à toutes les opérations du plan de données combinées.
Clés	Chaque Région prise en charge : 2 000	Oui	Le nombre maximum de clés que vous pouvez avoir dans ce compte dans la région actuelle, à l'exception des clés supprimées.

Historique du document pour le guide de l'utilisateur AWS de Payment Cryptography

Le tableau suivant décrit les versions de documentation relatives à la cryptographie des AWS paiements.

Modification	Description	Date
Nouvelle fonctionnalité - AS2805	Support aux algorithmes et aux flux pour soutenir le soutien AS2805 régional	17 décembre 2025
Nouvelle fonctionnalité - Réplication de clés dans plusieurs régions	Grâce à la réplication de clés multirégions, vous pouvez répliquer vos clés de cryptographie AWS de paiement sur plusieurs. Régions AWS	10 septembre 2025
Nouvelle fonctionnalité - ECDH	Avec cette version, l'ECDH peut être utilisé pour établir un KEK partagé pour un échange de clés ultérieur.	30 mars 2025
Nouvelles directives relatives à l'échange de clés	De nouvelles directives ont été fournies pour les principaux échanges. Des informations sur les commandes JCB courantes ont également été ajoutées.	31 janvier 2025
Lancement d'une nouvelle région	Points de terminaison supplémentaires pour le lancement dans de nouvelles régions en Europe (Francfort), en Europe (Irlande), en Asie-	31 juillet 2024

	Pacifique (Singapour) et en Asie-Pacifique (Tokyo)	
CloudTrail pour le plan de données et les clés dynamiques	Ajout d'informations sur l'utilisation CloudTrail pour les opérations (cryptographiques) sur le plan de données, y compris des exemples. Nous avons également ajouté des informations sur l'utilisation de clés dynamiques pour certaines fonctions afin de mieux prendre en charge les clés à usage unique ou à usage limité qui ne doivent pas être importées dans AWS Payment Cryptography	10 juillet 2024
Exemples mis à jour	Ajout de nouveaux exemples pour l'émission de cartes	1er juillet 2024
Publication de fonctionnalités	Ajout d'informations sur les points de terminaison VPC (PrivateLink) et exemples d'iCVV.	30 mai 2024
Publication de fonctionnalités	Informations ajoutées sur les nouvelles fonctionnalités relatives aux clés, à import/export l'utilisation de RSA et à l'exportation de clés DUKPT IPEK/IK .	15 janvier 2024
Première version	Publication initiale du guide de l'utilisateur de la cryptographie des AWS paiements	8 juin 2023

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.