Manual do usuário

AWS Payment Cryptography



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Payment Cryptography: Manual do usuário

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que é criptografia AWS de pagamento? 1
Conceitos 2
Terminologia do setor 4
Tipos de chaves comuns 4
Outros termos 8
Serviços relacionados 14
Para obter mais informações 14
Endpoints 14
Endpoints do ambiente de gerenciamento 15
Endpoints do plano de dados 18
Introdução 21
Pré-requisitos 21
Etapa 1: criar uma chave 22
Etapa 2: gerar um CVV2 valor usando a chave 23
Etapa 3: verificar o valor gerado na etapa 2 23
Etapa 4: realizar um teste negativo 24
Etapa 5 (opcional): limpeza 24
Managing keys 26
Criar chaves 26
Criação de uma chave de derivação básica TDES 2KEY 27
Criação de uma chave TDES 2KEY para CVV/CVV2 29
Criando uma chave HMAC 30
Criando uma AES-256 chave 31
Criando uma chave de criptografia PIN (PEK) 32
Criação de uma chave assimétrica (RSA) 33
Criando uma chave de valor de verificação de PIN (PVV) 34
Criação de uma chave ECC assimétrica 35
Chaves de listagem 36
Enabling and disabling keys 38
Iniciar o uso de chaves 38
Interromper o uso de chaves 40
Replicação de chaves 42
Benefícios da replicação de Multi-Region chaves 42
Como funciona Multi-Region a replicação de chaves 42

Limitações e considerações 42
Habilitando Multi-Region a replicação de chaves 44
Desativando a replicação de Multi-Region chaves 46
Considerações sobre segurança 47
Práticas recomendadas 47
Preços 47
Delete an key 48
Sobre o período de espera 49
Importando e exportando chaves 52
Importar chaves 54
Exportar chaves 79
Tópicos avançados 102
Usar aliases 121
Sobre aliases 122
Usar aliases em suas aplicações 125
APIs relacionadas 126
Obter chaves 126
key/certificate Associe o público a um par de chaves 128
Marcar chaves com tags 129
Sobre tags em AWS Payment Cryptography 129
Visualizar tags de chave no console 131
Gerenciar tags de chave com operações de API 131
Controlar o acesso às tags 134
Usar tags para controlar o acesso a chaves 138
Noções básicas sobre atributos de chave 142
Chaves simétricas 142
Chaves assimétricas 144
Operações de dados 146
Criptografe, descriptografe e recriptografe dados 146
Criptografar dados 147
Descriptografar dados 153
Gerar e verificar dados do cartão 157
Gerar dados do cartão 158
Verificar dados do cartão 159
Gerar, traduzir e verificar dados de PIN 161
Traduzir dados de PIN 162

Gerar dados de PIN ... 164
Verificar dados de PIN ... 168
Gere e verifique o ARQC ... 173
Gerar criptograma de solicitação de autenticação (ARQC) ... 173
Verificar o criptograma de solicitação de autenticação (ARQC) ... 176
Gerar e verificar MAC ... 180
Gerar MAC ... 182
Verificar MAC ... 186
Tipos de chaves para operações de dados específicas ... 188
GenerateCardData ... 189
VerifyCardData ... 190
GeneratePinData (para VISA/ABA esquemas) ... 191
GeneratePinData (para IBM3624) ... 192
VerifyPinData (para VISA/ABA esquemas) ... 193
VerifyPinData (para IBM3624) ... 194
Descriptografar dados ... 195
Criptografar dados ... 196
Traduzir dados de PIN ... 198
Generate/Verify MAC ... 199
GenerateMacEmvPinChange ... 200
VerifyAuthRequestCryptogram ... 202
Import/Export Chave ... 202
Tipos de chave não utilizados ... 203
Casos de uso comuns ... 204
Emissores e processadores de emissores ... 204
Funções gerais ... 204
Funções específicas da rede ... 238
Facilitadores de aquisição e pagamento ... 270
Usando teclas dinâmicas ... 271
Características específicas da região ... 274
AS2805 ... 274
Troca de chave inicial (KEK) ... 276
Validação do KEK ... 277
Criação e transmissão de chaves de trabalho ... 280
Exportação de chaves de trabalho ... 282
Tradução de Pin ... 283

Geração e validação de Mac 284
Segurança 285
Proteção de dados 286
Proteger material de chave 287
Criptografia de dados 287
Criptografia em repouso 287
Criptografia em trânsito 288
Privacidade do tráfego entre redes 288
Resiliência 289
Isolamento regional 289
Multi-tenant desenhar 290
Segurança da infraestrutura 291
Isolamento de hosts físicos 291
Use o Amazon VPC e o AWS PrivateLink 291
Considerações para AWS Endpoints VPC de criptografia de pagamento 292
Criação de um endpoint VPC para AWS Payment Cryptography 293
Conectar-se a um endpoint da VPC 294
Controlar o acesso a um endpoint da VPC 294
Usar um endpoint da VPC em uma declaração de política 298
Registrar o endpoint da VPC em log 302
TLS pós-quântico híbrido 304
Sobre o TLS pós-quântico 306
Como usar 306
Sobre a PQC 306
Como usar 306
Práticas recomendadas de segurança 307
Validação de conformidade 309
Conformidade do serviço 309
Conformidade com PI 310
Tópicos comuns 311
Escopo da avaliação 313
Operações de processamento de transações 315
Conformidade com P2PE 321
Gerenciamento de identidade e acesso 322
Público 322
Autenticação com identidades 323

Conta da AWS usuário-raiz 323
Usuários e grupos do IAM 323
Perfis do IAM 323
Gerenciar o acesso usando políticas 324
Identity-based políticas 324
Resource-based políticas 324
Listas de controle de acesso (ACLs) 325
Outros tipos de política 325
Vários tipos de política 325
Como AWS A criptografia de pagamento funciona com o IAM 326
AWS Políticas de criptografia Identity-based de pagamento 326
Autorização baseada em AWS Tags de criptografia de pagamento 328
Identity-based exemplos de políticas 328
Práticas recomendadas de política 329
Utilizar o console 330
Permitir que os usuários visualizem suas próprias permissões 330
Capacidade de acessar todos os aspectos do AWS Payment Cryptography 331
Capacidade de chamar APIs usando chaves especificadas 332
Capacidade de negar um recurso específico 333
Resource-based políticas 334
Considerações 335
Gerenciando políticas baseadas em recursos 336
Resource-based exemplos de políticas 337
Multi-party aprovação 339
Visão geral do 339
Operações protegidas 340
Pré-requisitos 340
Ativando e desativando o MPA 341
Introdução 342
Exemplo: importar um certificado raiz com o MPA ativado 342
AWS CloudTrail registro de eventos MPA 343
Verificando o status da solicitação e lidando com falhas 345
Solução de problemas 348
Monitoramento 349
CloudTrail troncos 349
AWS Informações de criptografia de pagamento em CloudTrail 350

Controle os eventos do avião em CloudTrail 351
Eventos de dados em CloudTrail 351
Compreendendo as entradas do arquivo de log do AWS Payment Cryptography Control Plane 352
Compreendendo as entradas do arquivo de log do plano de dados de criptografia de AWS pagamento 356
Detalhes criptográficos 359
Objetivos de projeto 360
Fundamentos 361
Primitivas criptográficas 361
Entropia e geração de números aleatórios 362
Operações de chave simétrica 362
Operações de chave assimétrica 362
Armazenamento de chaves 363
Importar chaves usando chaves simétricas 363
Importar chaves usando chaves assimétricas 363
Exportação de chaves 364
Protocolo de chave única derivada por transação (DUKPT) 364
Hierarquia de chaves 364
Operações internas 368
Proteção HSM 368
Gerenciamento geral de chaves 371
Gerenciamento de chaves de clientes 375
Segurança de comunicação 377
Registro em log e monitoramento 378
Operações do cliente 378
Gerar chaves 379
Importar chaves 379
Exportar chaves 380
Delete an key 381
Alternar chaves do 381
Cotas 382
Histórico do documento 384
..... ccclxxxvi

O que é criptografia AWS de pagamento?

AWS A criptografia de pagamento é um AWS serviço gerenciado que fornece acesso às funções criptográficas e ao gerenciamento de chaves usados no processamento de pagamentos de acordo com os padrões do setor de cartões de pagamento (PCI), sem a necessidade de adquirir instâncias HSM de pagamento dedicadas. AWS A criptografia de pagamento fornece aos clientes que realizam funções de pagamento, como adquirentes, facilitadores de pagamento, redes, comutadores, processadores e bancos, a capacidade de aproximar suas operações criptográficas de pagamento dos aplicativos na nuvem e minimizar as dependências de data centers auxiliares ou instalações de colocation contendo pagamento dedicado. HSMs

O serviço foi projetado para atender às regras aplicáveis do setor, incluindo PCI PIN, PCI P2PE e PCI DSS, e o serviço utiliza hardware com certificação PCI PTS HSM V3 e FIPS 140-2 Nível 3. Ele foi projetado para suportar baixa latência e altos níveis de disponibilidade e resiliência. AWS A criptografia de pagamento é totalmente elástica e elimina muitos dos requisitos operacionais locais HSMs, como a necessidade de provisionar hardware, gerenciar com segurança o material essencial e manter backups de emergência em instalações seguras. AWS A criptografia de pagamento também oferece a opção de compartilhar chaves eletronicamente com seus parceiros, eliminando a necessidade de compartilhar componentes de texto não criptografado em papel.

É possível usar a API do ambiente de gerenciamento de AWS Payment Cryptography para criar e gerenciar chaves.

É possível usar a API do plano de dados de AWS Payment Cryptography para usar chaves de criptografia para processamento de transações relacionadas a pagamentos e operações criptográficas associadas.

AWS A criptografia de pagamento fornece recursos importantes que você pode usar para gerenciar suas chaves:

- Crie e gerencie chaves de criptografia de AWS pagamento simétricas e assimétricas, incluindo chaves TDES, AES e RSA, e especifique a finalidade pretendida, como geração de CVV ou derivação de chaves DUKPT.
- Armazene automaticamente suas chaves AWS de criptografia de pagamento com segurança, protegidas por módulos de segurança de hardware (HSMs) e, ao mesmo tempo, imponha a separação de chaves entre os casos de uso.
- Crie, exclua, liste e atualize aliases, que são “nomes amigáveis” que podem ser usados para acessar ou controlar o acesso às suas chaves de criptografia AWS de pagamento.

- Marque suas chaves AWS de criptografia de pagamento para identificação, agrupamento, automação, controle de acesso e controle de custos.
- Importe e exporte chaves simétricas entre a criptografia de AWS pagamento e seu HSM (ou terceiros) usando chaves de criptografia de chave (KEK) seguindo a TR-31 (especificação interoperável de blocos de chaves de troca segura de chaves).
- Importe e exporte chaves de criptografia de chave simétricas (KEK) entre criptografia de AWS pagamento e outros sistemas usando pares de chaves assimétricas seguindo usando meios eletrônicos, como o TR-34 (Método para distribuição de chaves simétricas usando técnicas assimétricas).

Você pode usar suas chaves AWS de criptografia de pagamento em operações criptográficas, como:

- Criptografe, descriptografe e recriptografe dados com chaves de criptografia de pagamento simétricas ou assimétricas. AWS
- Traduzir com segurança dados confidenciais (como PINs do titular do cartão) entre chaves de criptografia sem expor o texto não criptografado de acordo com as regras de PCI PIN.
- Gere ou valide dados do titular do cartão, como CVV ou ARQC. CVV2
- Gerar e validar PINs do titular do cartão.
- Gerar ou validar assinaturas MAC.

Conceitos

Aprenda os termos e conceitos básicos usados na criptografia AWS de pagamento e como você pode usá-los para ajudar a proteger seus dados.

Alias

> Um nome fácil de usar associado a uma chave de criptografia AWS de pagamento. O alias pode ser usado de forma intercambiável com a chave ARN em muitas das operações da API de criptografia de pagamento. AWS Os aliases permitem que as chaves sejam alternadas ou alteradas sem afetar o código do aplicativo. O nome do alias é uma string com até 256 caracteres. Ele identifica de forma exclusiva uma chave de criptografia AWS de pagamento associada em uma conta e região. Na criptografia AWS de pagamento, os nomes de alias sempre começam com. `alias/`
>
> O formato de um nome de alias é o seguinte:

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

Por exemplo:

```
alias/sampleAlias2
```

ARN de chave

O ARN de chave é o nome do recurso da Amazon (ARN) de uma entrada de chave no AWS Payment Cryptography. É um identificador exclusivo e totalmente qualificado para a chave AWS de criptografia de pagamento. Um ARN de chave inclui uma região e um Conta da AWS ID gerado aleatoriamente. O ARN não está relacionado nem é derivado do material de chave. Como eles são atribuídos automaticamente durante as operações de criação ou importação, esses valores não são idempotentes. Importar a mesma chave várias vezes resultará em várias chaves ARNs com seu próprio ciclo de vida.

O formato de um ARN de chave é o seguinte:

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

Este é um exemplo de ARN de chave:

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

Identificador de chave

Um identificador de chave é uma referência a uma chave e uma (ou mais) delas são entradas típicas para operações de criptografia AWS de pagamento. Os identificadores de chave válidos podem ser um Key Arn ou um Key Alias.

AWS Chaves de criptografia de pagamento

AWS As chaves criptográficas de pagamento (chaves) são usadas para todas as funções criptográficas. As chaves são geradas diretamente por você usando o comando de criação de chaves ou adicionadas ao sistema ao chamar a importação de chaves. A origem de uma chave pode ser determinada pela análise do atributo KeyOrigin. AWS A criptografia de pagamento também suporta chaves derivadas ou intermediárias usadas durante operações criptográficas, como as usadas pela DUKPT.

Essas chaves possuem atributos imutáveis e mutáveis definidos na criação. Atributos como algoritmo, comprimento e uso são definidos na criação e não podem ser alterados. Outros, como data de vigência ou data de expiração, podem ser modificados. Consulte a Referência da API de criptografia de AWS pagamento para obter uma lista completa dos principais atributos da criptografia de AWS pagamento.

AWS As chaves criptográficas de pagamento têm tipos de chaves, definidos principalmente pelo ANSI X9 TR 31, que restringem seu uso à finalidade pretendida, conforme especificado no Requisito 19 do PIN PCI v3.1.

Os atributos são vinculados às chaves usando blocos de chaves quando armazenados, compartilhados com outras contas ou exportados conforme especificado no Requisito 18-3 do PCI PIN v3.1.

As chaves são identificadas na plataforma AWS de criptografia de pagamento usando um valor exclusivo conhecido como chave Amazon Resource Name (ARN).

> **Note**
>
> ARNA chave é gerada quando uma chave é inicialmente criada ou importada para o serviço AWS de criptografia de pagamento. Portanto, se você adicionar o mesmo material de chave várias vezes usando a funcionalidade de importação de chave, o mesmo material de chave estará localizado em várias chaves de ARNS, mas cada um com um ciclo de vida de chave diferente.

Terminologia do setor

Tópicos

- Tipos de chaves comuns
- Outros termos

Tipos de chaves comuns

AWS Chave de criptografia de pagamento

Uma chave criptográfica de AWS pagamento existe em uma única Região da AWS. Ele consiste em metadados e materiais importantes armazenados no AWS Payment Cryptography Service.

Uma chave pode ser importada de uma fonte externa como um bloco de TR-31 chaves ou gerada pelo AWS Payment Cryptography Service.

AWK

Uma chave de trabalho do adquirente (AWK) é uma chave normalmente usada para trocar dados entre um acquirer/acquirer processador e uma rede (como Visa ou Mastercard). Historicamente, a AWK utiliza 3DES para criptografia e seria representada como TR31_P0_PIN_ENCRYPTION_KEY.

BDK

Uma chave de derivação de base (BDK) é uma chave de trabalho usada para derivar chaves subsequentes e é comumente usada como parte do processo PCI PIN e PCI P2PE DUKPT. É indicada como TR31_B0_BASE_DERIVATION_KEY.

CMK

Uma chave mestra do cartão (CMK) é uma ou mais chaves específicas do cartão, normalmente derivadas de uma chave mestra do emissor, PAN e PSN e geralmente são chaves 3DES. Essas chaves são armazenadas no chip EMV durante a personalização. Exemplos de CMKs incluem chaves AC, SMI e SMC.

CMK-AC

Uma chave de criptograma de aplicativo (AC) é usada como parte das transações EMV para gerar o criptograma da transação e é um tipo de chave mestra do cartão.

CMK-SMI

Uma chave de integridade segura de mensagens (SIM) é usada como parte do EMV para verificar a integridade das cargas enviadas ao cartão usando MAC, como scripts de atualização de PIN. É um tipo de chave mestra de cartão.

CMK-SMC

Uma chave de confidencialidade segura de mensagens (SMC) é usada como parte do EMV para criptografar dados enviados ao cartão, como atualizações de PIN. É um tipo de chave mestra de cartão.

CVK

Uma chave de verificação de cartão (CVK) é uma chave usada para gerar CVV, CVV2 e valores similares usando um algoritmo definido, além de validar uma entrada. É indicada como TR31_C0_CARD_VERIFICATION_KEY.

IMK

Uma chave mestra do emissor (IMK) é uma chave mestra usada como parte da personalização do cartão com chip EMV. Normalmente, haverá 3 IMKs - um para cada chave AC (criptograma), SMI (chave mestra de script para integrity/signature) e SMC (chave mestra de script para). confidentiality/encryption

IK

Uma chave inicial (IK) é a primeira chave usada no processo DUKPT e deriva da Chave de Derivação Base (BDK). Nenhuma transação é processada nessa chave, mas ela é usada para derivar chaves futuras que serão usadas para transações. O método de derivação para criar um IK foi definido em X9.24-1:2017. Quando um TDES BDK é usado, X9.24-1:2009 é o padrão aplicável e o IK é substituído pela chave de criptografia de pino inicial (IPEK).

IPEK

Uma chave de criptografia PIN inicial (IPEK) é a chave inicial usada no processo de DUKPT e deriva da chave de derivação base (BDK). Nenhuma transação é processada nessa chave, mas ela é usada para derivar chaves futuras que serão usadas para transações. IPEK é um nome impróprio, pois essa chave também pode ser usada para derivar criptografia de dados e chaves mac. O método de derivação para criar um IPEK foi definido em. X9.24-1:2009 Quando um AES BDK é usado, X9.24-1:2017 é o padrão aplicável e o IPEK é substituído pela Chave Inicial (IK).

IWK

Uma chave de trabalho do emissor (IWK) é uma chave normalmente usada para trocar dados entre um issuer/issuer processador e uma rede (como Visa ou Mastercard). Historicamente, a IWK utiliza o 3DES para criptografia e é representada como TR31_P0_PIN_ENCRYPTION_KEY.

KBPK

Uma chave de proteção de bloco de chaves (KBPK) é um tipo de chave simétrica usada para proteger blocos de chaves e, portanto, outras chaves. wrap/encrypt Um KBPK é semelhante a um KEK, mas um KEK protege diretamente o material da chave, enquanto em TR-31 esquemas similares, o KBPK protege apenas indiretamente a chave de trabalho. Ao usar TR-31, TR31_K1_KEY_BLOCK_PROTECTION_KEY é o tipo de chave correto, embora TR31_K0_KEY_ENCRYPTION_KEY seja suportado de forma intercambiável para fins históricos.

KEK

Uma chave de criptografia de chave (KEK) é uma chave usada para criptografar outras chaves para transmissão ou armazenamento. As chaves destinadas a proteger outras chaves geralmente têm um KeyUsage TR31_K0_KEY_ENCRYPTION_KEY de acordo com o padrão. TR-31

PEK

Uma chave de criptografia de PIN (PEK) é um tipo de chave de trabalho usada para criptografar PINs para armazenamento ou transmissão entre duas partes. IWK e AWK são dois exemplos de usos específicos de chaves de criptografia de PINs. Essas chaves são representadas como TR31_P0_PIN_ENCRYPTION_KEY.

PGK

PGK (chave de geração de pinos) é outro nome para uma chave de verificação de pinos. Na verdade, não é usado para gerar pinos (que, por padrão, são números criptograficamente aleatórios), mas sim para gerar valores de verificação, como PVV.

PRK

A chave de região primária é a fonte de replicação autorizada para uma determinada chave de criptografia de pagamento para a qual a replicação foi ativada. O PRK é uma referência a um papel fundamental da criptografia de pagamento de origem em uma configuração de replicação de Multi-Region chaves. Quando a replicação é habilitada em uma chave de criptografia de pagamento, ela é chamada de PRK para essa configuração específica de replicação de chaves.

PVK

Uma chave de verificação de PIN (PVK) é um tipo de chave de trabalho usada para gerar valores de verificação de PIN, como PVV. Os dois tipos mais comuns são TR31_V1_IBM3624_PIN_VERIFICATION_KEY usado para gerar valores de deslocamento do IBM3624 e TR31_V2_VISA_PIN_VERIFICATION_KEY usado para valores de verificação. Visa/ABA Isso também pode ser conhecido como chave de geração de pinos.

RRK

As chaves de região da réplica são o material da chave replicada e os metadados copiados com segurança da PRK para uma réplica configurada. Região da AWS Um RRK é uma réplica somente para leitura de uma chave de criptografia de pagamento. O RRK é uma referência ao papel que uma chave específica desempenha em uma configuração de replicação de Multi-Region chaves. Todas as principais alterações nos metadados, incluindo as configurações de replicação, devem ser aplicadas ao PRK.

Outros termos

ARQC

O criptograma de solicitação de autorização (ARQC) é um criptograma gerado por um cartão com chip padrão EMV (ou implementação sem contato equivalente) no momento da transação. Normalmente, um ARQC é gerado por um cartão com chip e encaminhado a um emissor ou seu agente para verificação no momento da transação.

CVV

O valor de verificação do cartão é um valor secreto estático que era tradicionalmente incorporado em uma tarja magnética e usado para validar a autenticidade de uma transação. O algoritmo também é usado para outros fins, como iCVV, CAVV, CVV2. Ele pode não ser incorporado dessa forma para outros casos de uso.

CVV2

Um valor de verificação de cartão 2 é um valor secreto estático que era tradicionalmente impresso na frente (ou no verso) de um cartão de pagamento e é usado para verificar a autenticidade de pagamentos com cartão não presente (como por telefone ou on-line). Ele usa o mesmo algoritmo do CVV, mas o código do serviço está definido como 000.

iCVV

iCVV é um CVV2-like valor, mas incorporado aos dados equivalentes do track2 em um cartão EMV (Chip). Esse valor é calculado usando um código de serviço de 999 e é diferente do CVV1/CVV2 para evitar que informações roubadas sejam usadas para criar novas credenciais de pagamento de um tipo diferente. Por exemplo, se os dados da transação do chip foram obtidos, não é possível usar esses dados para gerar uma tarja magnética (CVV1) ou para compras on-line (CVV2).

Ele usa uma ??? chave

DUKPT

A chave única derivada por transação (DUKPT) é um padrão de gerenciamento de chaves normalmente usado para definir o uso de chaves de criptografia de uso único em formato físico. POS/POI Historicamente, a DUKPT utiliza 3DES para criptografia. O padrão da indústria para DUKPT é definido em ANSI. X9.24-3-2017

ECC

ECC (Elliptic Curve Cryptography) é um sistema de criptografia de chave pública que usa a matemática das curvas elípticas para criar chaves de criptografia. O ECC fornece o mesmo nível de segurança dos métodos tradicionais, como o RSA, mas com comprimentos de chave muito menores, fornecendo segurança equivalente de maneira mais eficiente. Isso é especialmente relevante para casos de uso em que o RSA não é uma solução prática (comprimento da chave RSA > 4096 bits). AWS A criptografia de pagamento suporta curvas definidas pelo NIST para uso em operações de ECDH.

ECDH

O ECDH (curva elíptica Diffie-Hellman) é um protocolo de acordo chave que permite que duas partes estabeleçam um segredo compartilhado (como um KEK ou um PEK). No ECDH, as Partes A e B têm seus próprios pares de chaves público-privadas e trocam chaves públicas entre si (na forma de certificados para criptografia de AWS pagamento), bem como metadados de derivação de chaves (método de derivação, tipo de hash e informações compartilhadas). Ambas as partes multiplicam sua chave privada pela chave pública da outra e, devido às propriedades da curva elíptica, ambas as partes são capazes de derivar (gerar) a chave resultante.

EMV

A EMV (originalmente Europay, Mastercard, Visa) é um órgão técnico que trabalha com as partes interessadas em pagamentos para criar padrões e tecnologias de pagamento interoperáveis. Um exemplo de padrão é para chip/contactless cartões e os terminais de pagamento com os quais eles interagem, incluindo a criptografia usada. A derivação de chave EMV se refere ao (s) método (s) de geração de chaves exclusivas para cada cartão de pagamento com base em um conjunto inicial de chaves, como um `IMK`

HSM

Um módulo de segurança de hardware (HSM) é um dispositivo físico que protege as operações criptográficas (por exemplo, criptografia, decodificação e assinaturas digitais), bem como as chaves subjacentes usadas para essas operações.

KCAAS

A Key Custodian As A Service (KCAAS) fornece uma variedade de serviços relacionados ao gerenciamento de chaves. Para chaves de pagamento, eles normalmente podem converter componentes-chave em papel em formulários eletrônicos suportados pela criptografia de AWS pagamento ou converter chaves protegidas eletronicamente em componentes em papel que podem ser exigidos por determinados fornecedores. Eles também podem fornecer serviços de

depósito de chaves para chaves cuja perda prejudicaria suas operações em andamento. Os fornecedores da KCAAS podem ajudar os clientes a aliviar a carga operacional de gerenciar materiais essenciais fora de um serviço seguro, como criptografia de AWS pagamento, de forma compatível com os padrões PCI DSS, PCI PIN e PCI P2PE. AWS A criptografia de pagamento oferece Troca de chaves físicas um recurso KCAAS integrado para converter os principais componentes em papel em formato eletrônico.

KCV

O valor de verificação de chave (KCV) se refere a uma variedade de métodos de soma de verificação usados principalmente para comparar as chaves entre si sem ter acesso ao material real da chave. O KCV também tem sido usado para validação de integridade (especialmente ao trocar chaves), embora essa função agora esteja incluída como parte de formatos de blocos de chaves, como TR-31. Para chaves TDES, o KCV é calculado criptografando 8 bytes, cada um com valor zero, com a chave a ser verificada e retendo os 3 bytes de ordem mais alta do resultado criptografado. Para chaves AES, o KCV é calculado usando um algoritmo CMAC em que os dados de entrada são 16 bytes de zero e retêm os 3 bytes de ordem mais alta do resultado criptografado. Para chaves HMAC, o KCV é calculado usando o hash selecionado na criação da chave em uma mensagem de tamanho zero, ocupando os 3 bytes mais à esquerda.

KDH

Um host de distribuição de chaves (KDH) é um dispositivo ou sistema que está enviando chaves em um processo de troca de chaves, como. TR-34 Ao enviar chaves do AWS Payment Cryptography, isso é considerado o KDH.

KIF

Uma instalação de injeção de chave (KIF) é um recurso seguro usado para inicializar terminais de pagamento, incluindo carregá-los com chaves de criptografia.

KRD

Um dispositivo de recebimento de chaves (KRD) é um dispositivo que está recebendo chaves em um processo de troca de chaves, como TR-34. Ao enviar chaves para criptografia AWS de pagamento, ela é considerada o KRD.

KSN

Um Número de Série da Chave (KSN) é um valor usado como entrada para o DUKPT encryption/decryption para criar chaves de criptografia exclusivas por transação. Normalmente, o KSN consiste em um identificador BDK, um ID de terminal semi-exclusivo e um contador de

transações, que é incrementado em cada transição processada em um determinado terminal de pagamento. Por exemplo X9.24, para o TDES, o KSN de 10 bytes normalmente consiste em 24 bits para o ID do conjunto de chaves, 19 bits para o ID do terminal e 21 bits para o contador de transações, embora o limite entre o ID do conjunto de chaves e o ID do terminal não tenha impacto na função da criptografia de pagamento. AWS Para o AES, o KSN de 12 bytes normalmente consiste em 32 bits para o ID do BDK, 32 bits para o identificador de derivação (ID) e 32 bits para o contador da transação.

MPoC

O mPOC (ponto de venda móvel em hardware comercial) é um padrão PCI que atende aos requisitos de segurança de soluções que permitem que os comerciantes aceitem PINs do titular do cartão ou pagamentos sem contato usando um smartphone ou outros dispositivos móveis comerciais prontos para uso (COTS).

PAN

Um número primário de conta (PAN) é um identificador exclusivo para uma conta, como um cartão de crédito ou débito. Normalmente, tem de 13 a 19 dígitos de comprimento. Os primeiros 6 a 8 dígitos identificam a rede e o banco emissor.

Bloco de PIN

Um bloco de dados contendo um PIN durante o processamento ou transmissão, bem como outros elementos de dados. Os formatos de bloco de PIN padronizam o conteúdo do bloco de PIN e como ele pode ser processado para recuperar o PIN. A maioria dos blocos de PIN é composta pelo PIN, pelo comprimento do PIN e frequentemente contém parte ou a totalidade do PAN. AWS A criptografia de pagamento suporta os formatos ISO 9564-1 0, 1, 3 e 4. O Formato 4 é necessário para chaves AES. Ao verificar ou traduzir PINs, é necessário especificar o bloco de PIN dos dados de entrada ou saída.

POI

O Ponto de Interação (POI), também frequentemente usado anonimamente com o Ponto de Venda (POS), é o dispositivo de hardware com o qual o titular do cartão interage para apresentar sua credencial de pagamento. Um exemplo de POI é o terminal físico em um estabelecimento comercial. Para obter a lista de terminais PCI PTS POI certificados, consulte o site da PCI.

PSN

O número de sequência PAN (PSN) é um valor numérico usado para diferenciar vários cartões emitidos com o mesmo PAN.

Chave pública

Ao usar cifras assimétricas (RSA, ECC), a chave pública é o componente público de um par de chaves público-privado. A chave pública pode ser compartilhada e distribuída para entidades que precisam criptografar dados para o proprietário do par de chaves público-privadas. Para operações de assinatura digital, a chave pública é usada para verificar a assinatura.

Chave privada

Ao usar cifras assimétricas (RSA, ECC), a chave privada é o componente privado de um par de chaves pública-privada. A chave privada é usada para descriptografar dados ou criar assinaturas digitais. Semelhante às chaves simétricas AWS de criptografia de pagamento, as chaves privadas são criadas com segurança pelos HSMs. Elas são descriptografadas somente na memória volátil do HSM e somente pelo tempo necessário para processar sua solicitação criptográfica.

PVV

Um valor de verificação de PIN (PVV) é um tipo de saída criptográfica que pode ser usada para verificar um pino sem armazenar o pino real. Embora seja um termo genérico, no contexto da criptografia de AWS pagamento, PVV se refere ao método Visa ou ABA PVV. Esse PVV é um número de quatro dígitos cujas entradas são o número do cartão, o número de sequência do pan, o próprio pan e uma chave de verificação do PIN. Durante o estágio de validação, a criptografia de AWS pagamento recria internamente o PVV usando os dados da transação e o compara novamente com o valor que foi armazenado pelo cliente da criptografia de pagamento. AWS Dessa forma, é conceitualmente semelhante a um hash criptográfico ou MAC.

RSA Wrap/Unwrap

O RSA wrap usa uma chave assimétrica para encapsular uma chave simétrica (como uma chave TDES) para transmissão para outro sistema. Somente o sistema com a chave privada correspondente pode descriptografar a carga e carregar a chave simétrica. Por outro lado, o RSA unwrap decodificará com segurança uma chave criptografada usando RSA e, em seguida, carregará a chave na criptografia de pagamento. AWS O RSA wrap é um método de troca de chaves de baixo nível e não transmite chaves no formato de bloco de chaves e não utiliza a assinatura de carga útil pela parte remetente. Controles alternativos devem ser considerados para verificar se a providência e os principais atributos não estão alterados.

TR-34 também utiliza o RSA internamente, mas é um formato separado e não é interoperável.

Código de serviço

Um código de serviço é um valor de três dígitos codificado na tarja magnética e no chip de um cartão de pagamento que define os atributos do cartão, como onde o cartão pode ser usado e se é necessária uma autorização adicional. O código de serviço é usado como entrada ao calcular os valores de verificação. Valores de código de serviço diferentes produzem diferentes tipos de valores de verificação, como CVV (usando o código de serviço da tarja magnética do cartão), iCVV (código de serviço 999 para cartões com chip) e CVV2 (código de serviço 000).

TR-31

TR-31 (formalmente definido como ANSI X9 TR 31) é um formato de bloco de chave definido pelo American National Standards Institute (ANSI) para dar suporte à definição de atributos-chave na mesma estrutura de dados dos próprios dados-chave. O formato do bloco de TR-31 chaves define um conjunto de atributos-chave vinculados à chave para que sejam mantidos juntos. AWS A criptografia de pagamento usa termos TR-31 padronizados sempre que possível para garantir a separação adequada das chaves e a finalidade da chave. TR-31 foi substituído pelo ANSI. X9.143-2022

TR-34

TR-34 é uma implementação do ANSI X9.24-2 que descreveu um protocolo para distribuir chaves simétricas com segurança (como 3DES e AES) usando técnicas assimétricas (como RSA). AWS A criptografia de pagamento usa TR-34 métodos para permitir a importação e exportação seguras de chaves.

X9.143

X9.143 é um formato de bloco de chaves definido pelo American National Standards Institute (ANSI) para oferecer suporte à proteção de uma chave e de atributos chave na mesma estrutura de dados. O formato do bloco de chaves define um conjunto de atributos-chave vinculados à chave para que sejam mantidos juntos. AWS A criptografia de pagamento usa termos X9.143 padronizados sempre que possível para garantir a separação adequada das chaves e a finalidade da chave. X9.143 substitui a TR-31proposta anterior, embora na maioria dos casos sejam compatíveis com versões anteriores e anteriores e os termos sejam frequentemente usados de forma intercambiável.

Serviços relacionados

AWS Key Management Service

AWS O Key Management Service (AWS KMS) é um serviço gerenciado que facilita a criação e o controle das chaves criptográficas usadas para proteger seus dados. AWS O KMS usa módulos de segurança de hardware (HSMs) para proteger e validar suas chaves AWS KMS.

AWS CloudHSM

AWS CloudHSM fornece aos clientes instâncias HSM dedicadas de uso geral na AWS nuvem. AWS CloudHSM pode fornecer uma variedade de funções criptográficas, como criação de chaves, assinatura de dados ou criptografia e descriptografia de dados.

Para obter mais informações

- Para saber mais sobre os termos e conceitos usados na criptografia AWS de pagamento, consulte Conceitos AWS de criptografia de pagamento.
- Para obter informações sobre a API AWS Payment Cryptography Control Plane, consulte Referência da API AWS Payment Cryptography Control Plane.
- Para obter informações sobre a API AWS Payment Cryptography Data Plane, consulte Referência da API AWS Payment Cryptography Data Plane.
- Para obter informações técnicas detalhadas sobre como a criptografia AWS de pagamento usa criptografia e protege as chaves de criptografia de AWS pagamento, consulte Detalhes criptográficos.

Endpoints para AWS Payment Cryptography

Para se conectar programaticamente AWS Payment Cryptography, você usa um endpoint, a URL do ponto de entrada do serviço. Os AWS SDKs e as ferramentas de linha de comando usam automaticamente o endpoint padrão do serviço Região da AWS com base no contexto regional de uma solicitação, portanto, normalmente não há necessidade de definir explicitamente esses valores. Quando necessário, você pode especificar um endpoint diferente para suas solicitações de API.

Endpoints do ambiente de gerenciamento

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA (Ohio)	us-east-2	controlplane.payment-cryptography.us-east-2.amazonaws.com controlplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS	
Leste dos EUA (Norte da Virgínia)	us-east-1	controlplane.payment-cryptography.us-east-1.amazonaws.com controlplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS	
Oeste dos EUA (Oregon)	us-west-2	controlplane.payment-cryptography.us-west-2.amazonaws.com controlplane.payment-cryptography.us-west-2.api.aws	HTTPS HTTPS	
África (Cidade do Cabo)	af-south-1	controlplane.payment-cryptography.af-south-1.amazonaws.com controlplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS	
Ásia-Pacífico (Hyderabad)	ap-south-2	controlplane.payment-cryptography.ap-south-2.amazonaws.com controlplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS	
Ásia-Pacífico (Mumbai)	ap-south-1	controlplane.payment-cryptography.ap-south-1.amazonaws.com	HTTPS HTTPS	

Nome da região	Região	Endpoint	Protocolo
		controlplane.payment-cryptography.ap-south-1.api.aws	
Ásia-Pacífico (Osaka)	ap-northeast-3	controlplane.payment-cryptography.ap-northeast-3.amazonaws.com controlplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS
Ásia-Pacífico (Singapura)	ap-southeast-1	controlplane.payment-cryptography.ap-southeast-1.amazonaws.com controlplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS
Ásia-Pacífico (Sydney)	ap-southeast-2	controlplane.payment-cryptography.ap-southeast-2.amazonaws.com controlplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS
Ásia-Pacífico (Tóquio)	ap-northeast-1	controlplane.payment-cryptography.ap-northeast-1.amazonaws.com controlplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS
Canadá (Central)	ca-central-1	controlplane.payment-cryptography.ca-central-1.amazonaws.com controlplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS

Nome da região	Região	Endpoint	Protocolo	
Europa (Frankfurt)	eu-central-1	controlplane.payment-cryptography.eu-central-1.amazonaws.com controlplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS	
Europa (Irlanda)	eu-west-1	controlplane.payment-cryptography.eu-west-1.amazonaws.com controlplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS	
Europa (Londres)	eu-west-2	controlplane.payment-cryptography.eu-west-2.amazonaws.com controlplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS	
Europa (Paris)	eu-west-3	controlplane.payment-cryptography.eu-west-3.amazonaws.com controlplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS	
América do Sul (São Paulo)	sa-east-1	controlplane.payment-cryptography.sa-east-1.amazonaws.com controlplane.payment-cryptography.sa-east-1.api.aws	HTTPS HTTPS	

Endpoints do plano de dados

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA (Ohio)	us-east-2	dataplane.payment-cryptography.us-east-2.amazonaws.com dataplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS	
Leste dos EUA (Norte da Virgínia)	us-east-1	dataplane.payment-cryptography.us-east-1.amazonaws.com dataplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS	
Oeste dos EUA (Oregon)	us-west-2	dataplane.payment-cryptography.us-west-2.amazonaws.com dataplane.payment-cryptography.us-west-2.api.aws	HTTPS HTTPS	
África (Cidade do Cabo)	af-south-1	dataplane.payment-cryptography.af-south-1.amazonaws.com dataplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS	
Ásia-Pacífico (Hyderabad)	ap-south-2	dataplane.payment-cryptography.ap-south-2.amazonaws.com dataplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS	
Ásia-Pacífico (Mumbai)	ap-south-1	dataplane.payment-cryptography.ap-south-1.amazonaws.com	HTTPS HTTPS	

Nome da região	Região	Endpoint	Protocolo
		dataplane.payment-cryptography.ap-south-1.api.aws	
Ásia-Pacífico (Osaka)	ap-northeast-3	dataplane.payment-cryptography.ap-northeast-3.amazonaws.com dataplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS
Ásia-Pacífico (Singapura)	ap-southeast-1	dataplane.payment-cryptography.ap-southeast-1.amazonaws.com dataplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS
Ásia-Pacífico (Sydney)	ap-southeast-2	dataplane.payment-cryptography.ap-southeast-2.amazonaws.com dataplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS
Ásia-Pacífico (Tóquio)	ap-northeast-1	dataplane.payment-cryptography.ap-northeast-1.amazonaws.com dataplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS
Canadá (Central)	ca-central-1	dataplane.payment-cryptography.ca-central-1.amazonaws.com dataplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS

Nome da região	Região	Endpoint	Protocolo	
Europa (Frankfurt)	eu-central-1	dataplane.payment-cryptography.eu-central-1.amazonaws.com dataplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS	
Europa (Irlanda)	eu-west-1	dataplane.payment-cryptography.eu-west-1.amazonaws.com dataplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS	
Europa (Londres)	eu-west-2	dataplane.payment-cryptography.eu-west-2.amazonaws.com dataplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS	
Europa (Paris)	eu-west-3	dataplane.payment-cryptography.eu-west-3.amazonaws.com dataplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS	
América do Sul (São Paulo)	sa-east-1	dataplane.payment-cryptography.sa-east-1.amazonaws.com dataplane.payment-cryptography.sa-east-1.api.aws	HTTPS HTTPS	

Introdução à criptografia AWS de pagamento

Para começar a usar a criptografia de AWS pagamento, primeiro você deve criar chaves e depois usá-las em várias operações criptográficas. O tutorial abaixo fornece um caso de uso simples para gerar uma chave a ser usada para generating/verifying CVV2 valores. Para experimentar outros exemplos e explorar padrões de implantação na AWS, experimente o seguinte workshop sobre criptografia de AWS pagamentos ou explore nosso projeto de amostra disponível em GitHub

Este tutorial explica como criar uma chave única e realizar operações criptográficas usando a chave. Depois disso, você exclui a chave se não tiver uso para ela, o que completa o ciclo de vida da chave.

> **Warning**
>
> Os exemplos deste guia do usuário podem usar valores de amostra. É altamente recomendável não usar valores de amostra em um ambiente de produção, como números de série de chaves.

Tópicos

- Pré-requisitos
- Etapa 1: criar uma chave
- Etapa 2: gerar um CVV2 valor usando a chave
- Etapa 3: verificar o valor gerado na etapa 2
- Etapa 4: realizar um teste negativo
- Etapa 5 (opcional): limpeza

Pré-requisitos

Antes de começar, verifique se:

- Você tem permissão para acessar o serviço. Para obter mais informações, consulte as políticas do IAM.
- Você tem o AWS CLI instalado. Você também pode usar AWS SDKsou acessar AWS APIsa Criptografia de AWS Pagamento, mas as instruções deste tutorial usam o. AWS CLI

Etapa 1: criar uma chave

A primeira etapa é criar uma chave. Neste tutorial, você cria uma chave CVK 3DES de comprimento duplo (2KEY TDES) para gerar e verificar valores CVV/. CVV2

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

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
    }
}
```

Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi. Isso será necessário na próxima etapa.

Etapa 2: gerar um CVV2 valor usando a chave

Nesta etapa, você gera uma CVV2 para uma determinada data PAN de validade usando a chave da etapa 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"
}
```

Anote o `cardDataValue`; neste caso, o número 144, de 3 dígitos. Isso será necessário na próxima etapa.

Etapa 3: verificar o valor gerado na etapa 2

Neste exemplo, você valida a CVV2 partir da etapa 2 usando a chave que você criou na etapa 1.

Execute o comando a seguir para validar o. 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"
}
```

O serviço retorna uma resposta HTTP de 200 para indicar que validou o. CVV2

Etapa 4: realizar um teste negativo

Nesta etapa, você cria um teste negativo em que o não CVV2 está correto e não é validado. Você tenta validar um erro CVV2 usando a chave criada na etapa 1. Essa é uma operação esperada, por exemplo, se o titular do cartão digitou o erro CVV2 na finalização da compra.

```
$  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.
```

O serviço retorna uma resposta HTTP de 400 com a mensagem “Falha na verificação dos dados de validação do cartão” e um motivo INVALID_VALIDATION_DATA.

Etapa 5 (opcional): limpeza

Agora, você pode excluir a chave criada na etapa 1. Para minimizar as alterações irrecuperáveis, o período padrão da exclusão da chave é de sete dias.

```
$ 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"
    }
}
```

Observe os dois campos na saída. Por padrão, `deletePendingTimestamp` é definido para sete dias no futuro. O keyState está definido como `DELETE_PENDING`. Você pode cancelar esta exclusão a qualquer momento antes do horário programado ao chamar `restore-key`.

Managing keys

Para começar a usar a criptografia AWS de pagamento, crie uma chave de criptografia AWS de pagamento.

Esta seção explica como criar e gerenciar vários tipos de chaves de criptografia AWS de pagamento ao longo de seu ciclo de vida. Você aprenderá a criar, visualizar e editar chaves, bem como a marcar chaves, criar aliases de chave e ativar ou desativar chaves.

Uma chave AWS de criptografia de pagamento é um recurso regional. Se você pretende usar uma determinada chave em várias Regiões da AWS, você pode ativar a replicação de Multi-Region chaves, que copia com segurança o material e os metadados da chave para Regiões da AWS você especificar na mesma AWS partição e conta. A chave de origem na replicação de Multi-Region chaves é conhecida como chave de região primária (PRK) e continua sendo a fonte oficial para todas as atividades de gerenciamento de chaves. A chave replicada é conhecida como chave de região de réplica (RRK) e é uma réplica somente para leitura da PRK. Você deve considerar o uso de Multi-Region chaves com suas chaves para atingir as metas de design relacionadas à disponibilidade, recuperação de desastres e baixa latência.

Tópicos

- Criar chaves
- Chaves de listagem
- Enabling and disabling keys
- Replicação AWS Chaves de criptografia de pagamento
- Delete an key
- Importando e exportando chaves
- Usar aliases
- Obter chaves
- Marcar chaves com tags
- Compreendendo os principais atributos do AWS Chave de criptografia de pagamento

Criar chaves

Você pode criar chaves AWS de criptografia de pagamento usando a operação da CreateKey API. Ao criar uma chave, você especifica atributos como o algoritmo da chave, o uso da chave, as

operações permitidas e se ela é exportável. Você não pode alterar essas propriedades depois de criar a chave AWS de criptografia de pagamento.

> **Note**
>
> Se a replicação de Multi-Region chaves estiver habilitada para você Conta da AWS e você criar uma chave de criptografia de pagamento, essa chave se tornará automaticamente uma chave de região primária (PRK). O PRK é replicado mesmo que você não especifique o `--replication-regions` parâmetro no CreateKey comando. Para obter mais informações, consulte Como funciona Multi-Region a replicação de chaves.

Exemplos

- Criação de uma chave de derivação básica TDES 2KEY
- Criação de uma chave TDES 2KEY para CVV/CVV2
- Criando uma chave HMAC
- Criando uma AES-256 chave
- Criando uma chave de criptografia PIN (PEK)
- Criação de uma chave assimétrica (RSA)
- Criando uma chave de valor de verificação de PIN (PVV)
- Criação de uma chave ECC assimétrica

Criação de uma chave de derivação básica TDES 2KEY

> **Note**
>
> Embora o 3KEY TDES seja compatível com KeyUsage = TR31_B0_BASE_DERIVATION_KEY, nenhuma operação atualmente oferece suporte a esse tipo de chave. O X9.24 padrão ANSI (DUKPT) define suporte somente para os tipos de chave TDES_2KEY e AES.

Example

Esse comando cria uma chave de derivação 2KEY TDES para o DUKPT que será replicada das regiões Oeste dos EUA (Oregon) para Leste dos EUA (Ohio). A resposta inclui os parâmetros da

solicitação, um nome de recurso da Amazon (ARN) para chamadas subsequentes e um Key Check Value (KCV).

```
$ aws payment-cryptography create-key --exportable --key-attributes \
     "KeyUsage=TR31_B0_BASE_DERIVATION_KEY, \
     KeyClass=SYMMETRIC_KEY,KeyAlgorithm=TDES_2KEY, \
     KeyModesOfUse={DeriveKey=true}" \
     --replication-regions us-east-2 --region us-west-2
```

Resultado do exemplo:

```
{
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": true,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": false,
                "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"
}
```

Criação de uma chave TDES 2KEY para CVV/CVV2

Example

Esse comando cria uma chave TDES 2KEY para gerar e verificar CVV/CVV2 valores. A resposta inclui os parâmetros da solicitação, um nome de recurso da Amazon (ARN) para chamadas subsequentes e um 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}'
```

Resultado do exemplo:

```
{
    "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"
    }
}
```

Criando uma chave HMAC

Example

As chaves HMAC são usadas para gerar ou verificar códigos de autenticação de mensagens hash (HMAC). Com as chaves HMAC, o tipo de hash é atribuído no momento da criação da chave (como HMAC_SHA224 e HMAC_SHA512) e não pode ser modificado.

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

Resultado do exemplo:

```
{
 "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"
 }
}
```

Criando uma AES-256 chave

Example

Esse comando cria uma chave AES-256 simétrica para criptografia e descriptografia de dados. As chaves AES fornecem criptografia forte para dados confidenciais e são comumente usadas no processamento de pagamentos para criptografar dados do titular do cartão e outras informações confidenciais. No entanto, o TDES é mais comumente usado em casos de uso do emissor, como o EMV.

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

Resultado do exemplo:

```
{
    "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"
    }
}
```

Criando uma chave de criptografia PIN (PEK)

Example

Esse comando cria uma chave TDES de 3 teclas para criptografar valores de PIN, embora as chaves de pino também possam ser AES, dependendo da sua necessidade de interoperabilidade. Você pode usar essa chave para armazenar PINs com segurança ou descriptografar PINs durante a verificação, como em uma transação. A resposta inclui os parâmetros da solicitação, um ARN para chamadas subsequentes e um 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}'
```

Resultado do exemplo:

```
{
    "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"
    }
}
```

Criação de uma chave assimétrica (RSA)

Example

Esse comando gera um novo par de chaves RSA de 2048 bits assimétrico. Ele cria uma nova chave privada e sua chave pública correspondente. Você pode recuperar a chave pública usando a PublicCertificate API 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}'
```

Resultado do exemplo:

```
{
    "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"
    }
}
```

Criando uma chave de valor de verificação de PIN (PVV)

Example

Esse comando cria uma chave TDES 3KEY para gerar valores de PVV. Você pode usar essa chave para gerar um PVV que pode ser comparado com um PVV calculado posteriormente. A resposta inclui os parâmetros da solicitação, um ARN para chamadas subsequentes e um 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}'
```

Resultado do exemplo:

```
{
    "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"
    }
}
```

Criação de uma chave ECC assimétrica

Exemplo

Esse comando gera um par de chaves ECC para estabelecer um acordo de chave ECDH (curva elíptica Diffie-Hellman) entre duas partes. Com o ECDH, cada parte gera seu próprio par de chaves ECC com a finalidade da chave K3 e o modo de uso X, e eles trocam chaves públicas. Em seguida, ambas as partes usam sua chave privada e a chave pública recebida para estabelecer uma chave derivada compartilhada.

Para manter o princípio de uso único das chaves criptográficas em pagamentos, recomendamos não reutilizar pares de chaves ECC para várias finalidades, como derivação e assinatura de chaves 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}'
```

Resultado do exemplo

```
{
    "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"
    }
}
```

Chaves de listagem

Use a ListKeys operação para obter uma lista de chaves acessíveis para você em sua conta e região.

Example

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

Resultado do exemplo:

```
{
    "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"
        }
    ]
}
```

Enabling and disabling keys

Você pode desativar e reativar as chaves AWS de criptografia de pagamento. Quando uma chave é criada, ela é habilitada por padrão. Se você desabilitar uma chave, ela não poderá ser usada em nenhuma operação criptográfica até que você a reative. Start/stop os comandos de uso têm efeito imediato, portanto, é recomendável que você revise o uso antes de fazer essa alteração. Também é possível definir uma alteração (iniciar ou interromper o uso) para entrar em vigor no futuro usando o parâmetro opcional `timestamp`.

Por ser temporária e facilmente desfeita, desativar uma chave de criptografia de AWS pagamento é uma alternativa mais segura do que excluir uma chave de criptografia de AWS pagamento, uma ação que é destrutiva e irreversível. Se você estiver pensando em excluir uma chave de criptografia de AWS pagamento, desative-a primeiro e certifique-se de que não precisará usar a chave para criptografar ou descriptografar dados no futuro.

Tópicos

- Iniciar o uso de chaves
- Interromper o uso de chaves

Iniciar o uso de chaves

O uso de chaves deve ser habilitado para que seja possível usar uma chave para operações criptográficas. Se uma chave não estiver ativada, você usará essa operação para torná-la utilizável. O campo `UsageStartTimestamp` representará quando a chave became/will se tornar ativa. Isso acontecerá no passado para um token ativado e no futuro se a ativação estiver pendente.

Example

Neste exemplo, solicita-se que uma chave seja habilitada para que seja usada. A resposta inclui as principais informações e o sinalizador de ativação foi alterado para verdadeiro. Isso também será refletido no objeto de resposta 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"
  }
}
```

Interromper o uso de chaves

Se você não planeja mais usar uma chave, pode interromper o uso dela para evitar mais operações criptográficas. Essa operação não é permanente e você pode revertê-la usando Iniciar o uso de chaves. Também é possível definir uma chave para ser desativada no futuro. O campo `UsageStopTimestamp` representará quando a chave for became/will desativada.

Example

Neste exemplo, é solicitado que você interrompa o uso da chave no futuro. Após a execução, essa chave não pode ser usada para operações criptográficas, a menos que seja reativada por meio da opção Iniciar o uso de chaves. A resposta inclui as informações da chave e o sinalizador de ativação foi alterado para falso. Isso também será refletido no objeto de resposta 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"
  }
}
```

Replicação AWS Chaves de criptografia de pagamento

AWS A criptografia de pagamento suporta a replicação de Multi-Region chaves, permitindo que você distribua com segurança o material e os metadados da chave de qualquer chave criptográfica de AWS pagamento para uma ou mais na Regiões da AWS mesma partição e conta. AWS

A chave de origem é conhecida como chave de região primária (PRK) e continua sendo a fonte autorizada de todas as atividades de gerenciamento de chaves, enquanto as chaves de região de réplica (RRK) podem ser usadas para operações criptográficas em suas respectivas regiões. Regiões da AWS

Benefícios da replicação de Multi-Region chaves

A seguir, descrevemos alguns benefícios da replicação de Multi-Region chaves.

- Configuração mais fácil para aplicativos de alta disponibilidade - A criptografia de AWS pagamento gerencia a distribuição de chaves para que você possa usar uma chave em várias Regiões da AWS sem precisar criar cópias dissociadas de uma determinada chave.
- Chaves de alta disponibilidade e baixa latência - Com Multi-Region a replicação de chaves, você pode acessar suas chaves em várias, Regiões da AWS tornando-as altamente disponíveis, resultando em menor latência.
- Durabilidade do material chave - as chaves da região de réplica são réplicas completas de chaves e podem ser usadas independentemente da chave da região primária em operações criptográficas. Um RRK fornece uma réplica durável no caso de uma perda catastrófica de dados de um PRK.

Como funciona Multi-Region a replicação de chaves

Quando a replicação de Multi-Region chaves está ativada, o serviço de criptografia de AWS pagamento usa mecanismos seguros de distribuição de chaves para copiar o material e os metadados da chave para a réplica Regiões da AWS que você especificar. As alterações nos metadados da chave de uma região primária, como atributos, estado e habilitação da chave, são automaticamente replicadas para as chaves da região de réplica.

Limitações e considerações

A seguir estão algumas das Multi-Region principais limitações e considerações da replicação.

- As chaves de criptografia de pagamento devem ser definidas como exportáveis para usar a replicação de Multi-Region chaves.
- Você deve ativar esse recurso para uma chave de criptografia de pagamento Região da AWS ou para uma chave específica.
  - Se esse recurso estiver ativado para um Região da AWS, todas as chaves AWS de criptografia de pagamento criadas após a ativação serão replicadas para o especificado. Região da AWS As chaves criadas nessa região se tornarão chaves da região primária. As chaves existentes nessa região não serão replicadas automaticamente. Você pode ativar Multi-Region a replicação de chaves existentes Região da AWS em um nível de chave.
  - Cada um Região da AWS pode ter configurações exclusivas de replicação de Multi-Region chaves.
  - As configurações de Multi-Region replicação de uma chave têm precedência sobre a configuração de replicação de Região da AWS Multi-Region chaves.
- Uma chave de região de réplica não pode ser configurada para ser replicada para outra. Regiões da AWS
- Multi-Region a replicação de chaves está disponível para chaves simétricas de criptografia de pagamento, como Triple Data Encryption Standard (3DES), Advanced Encryption Standard (AES) e Hash-based Message Authentication Code (HMAC), bem como chaves de criptografia de pagamento assimétricas, como RSA e Elliptic Curve Cryptography (ECC).
- As chaves da região de réplica são chaves somente para leitura. Todas as alterações na chave da região primária serão aplicadas às chaves da região de réplica.
- Eventualmente, as alterações nas chaves da região primária são consistentes com as chaves da região de réplica.
- As chaves de criptografia de pagamento só podem ser replicadas com a mesma AWS partição e conta.
- As chaves da região de réplica contam para seu limite Conta da AWS de criptografia AWS de pagamento de nível.
- A chave da região primária e a chave da região de réplica usam o mesmo identificador de chave, o que permite referenciar as duas chaves pelo mesmo ARN nas políticas do IAM.
- Você precisa ter `CreateKey` permissões na réplica Região da AWS para que a replicação seja bem-sucedida.

Habilitando Multi-Region a replicação de chaves

Há duas maneiras de ativar a replicação de Multi-Region chaves para chaves de criptografia AWS de pagamento.

1. Região da AWS: a replicação de Multi-Region chaves é aplicada a todas as novas chaves criadas Região da AWS quando ativada. Esse método fornece replicação consistente para todas as chaves.
2. Chaves AWS de criptografia de pagamento específicas: você pode gerenciar Multi-Region a replicação de chaves individuais, permitindo um nível de controle mais granular.

Quando a replicação de Multi-Region chaves estiver ativada, suas chaves de criptografia de pagamento serão replicadas de acordo com o Regiões da AWS especificado por você.

> **Important**
>
> Multi-Region a replicação de chaves não pode ser pausada. Suas chaves são automaticamente replicadas para o especificado por Regiões da AWS você quando a replicação é ativada. Multi-Region a replicação de chaves pode ser desativada para chaves específicas Região da AWS ou de criptografia de pagamento. Você deve remover o Região da AWS como uma região de replicação da chave da Região Primária para excluir a chave da Região de Réplica.
> Como alternativa, você pode chamar o comando `StopKeyUsage` API ou stop-key-usage CLI em seu PRK para interromper o uso do PRK e de todos os RRKs associados. Você não poderá usar essas chaves em operações criptográficas. O uso `StopKeyUsage` do comando API ou stop-key-usage CLI não interromperá a replicação contínua de Multi-Region chaves habilitada para sua PRK.

Você pode verificar as configurações de replicação de Multi-Region chaves para chaves AWS de criptografia de pagamento em um determinado momento Região da AWS chamando o comando `GetDefaultKeyReplicationRegions` API ou get-default-key-replication-regions CLI. As chaves no Região da AWS local em que você chama essa ação ou comando da API se tornarão sua PRK.

Use os procedimentos a seguir para ativar a replicação de Multi-Region chaves.

For Região da AWS

- Use o comando a seguir para ativar a replicação de Multi-Region chaves para um Região da AWS que você especificar. Neste exemplo, a replicação de Multi-Region chaves está habilitada no Leste dos EUA (Ohio) e no Oeste dos EUA (Oregon). Para usar esse comando, substitua o comando *italicized placeholder text* no exemplo por suas próprias informações.

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

> **Note**
>
> Habilitar a replicação de Multi-Region chaves para um não Região da AWS alterará a configuração de replicação de nenhuma chave de criptografia de AWS pagamento existente. Você pode ativar esse recurso para chaves existentes no nível da chave. Somente as chaves criadas após a replicação de Multi-Region chaves ser ativada para e Região da AWS usarão as configurações de replicação da região.

For specific AWS Payment Cryptography keys

- Use o comando a seguir para ativar a replicação de Multi-Region chaves de criptografia de pagamento específicas. Neste exemplo, a replicação de Multi-Region chaves está habilitada no Leste dos EUA (Ohio). Para usar esse comando, substitua o comando *italicized placeholder text* no exemplo por suas próprias informações.

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

Como alternativa, você pode criar uma nova chave de criptografia de pagamento com esse recurso ativado, incluindo a replicação Regiões da AWS em sua solicitação de criação de chave.

> **Note**
>
> As principais configurações de replicação têm precedência sobre a configuração de Região da AWS replicação.

Desativando a replicação de Multi-Region chaves

Se quiser desabilitar a replicação de Multi-Region chaves, você pode chamar os comandos disable-default-key-replication ou remove-key-replication-regions CLI, dependendo de como a replicação de Multi-Region chaves está habilitada. Você precisará especificar o ARN da chave e o Região da AWS para desativar a replicação da Multi-Region chave.

Considerações

Eventualmente, as exclusões de chaves da região de replicação são consistentes.

Você pode verificar as configurações de replicação de Multi-Region chaves para chaves AWS de criptografia de pagamento em um determinado momento Região da AWS chamando o comando `GetDefaultKeyReplicationRegions` API ou get-default-key-replication-regions CLI.

Use os procedimentos a seguir para desativar a replicação de Multi-Region chaves.

For Região da AWS

- Use o comando a seguir para desativar a replicação de Multi-Region chaves para um Região da AWS que você especificar. Neste exemplo, a replicação de Multi-Region chaves está desativada no Leste dos EUA (Ohio). Para usar esse comando, substitua o comando *`italicized placeholder text`* no exemplo por suas próprias informações.

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

For specific AWS Payment Cryptography keys

- Use o comando a seguir para desativar a replicação de chaves para uma Multi-Region chave específica de criptografia de pagamento. Neste exemplo, a replicação de Multi-Region chaves está sendo desativada no Leste dos EUA (Ohio). Para usar esse comando, substitua o comando *`italicized placeholder text`* no exemplo por suas próprias informações.

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

Considerações sobre segurança

A seguir estão as considerações de segurança ao usar a replicação de Multi-Region chaves para suas chaves de criptografia de pagamento. Para obter mais informações, consulte Práticas recomendadas de segurança para AWS Payment Cryptography.

- Limite o compartilhamento de materiais importantes.
- Siga o princípio das permissões com menos privilégios ao criar políticas do IAM.
- Você não pode fazer alterações na chave da região de réplica, pois é uma chave somente para leitura.

Práticas recomendadas

A seguir estão algumas das melhores práticas ao usar a replicação de Multi-Region chaves com chaves AWS de criptografia de pagamento.

- Garanta que seu aplicativo continue funcionando mesmo que a replicação da Multi-Region chave para o especificado não Região da AWS seja imediata. Se precisar saber quando a replicação da Multi-Region chave está concluída, você pode monitorar com a ação da GetKey API. Você pode monitorar os principais eventos de replicação com AWS CloudTrail.
- Teste e implemente processos de implantação automatizados em caso de falha de uma região Região da AWS para outra.

Preços

Você é cobrado pelas chaves de região de réplica criadas com a criptografia AWS de pagamento. Essas chaves são cobradas por Região da AWS. Para obter as informações mais recentes sobre preços de criptografia de pagamento, consulte a página de preços AWS de criptografia de pagamento.

Delete an key

A exclusão de uma chave de criptografia de AWS pagamento exclui o material da chave e todos os metadados associados à chave e é irreversível, a menos que uma cópia da chave esteja disponível fora da criptografia de pagamento. AWS Após a exclusão de uma chave, não é mais possível descriptografar os dados criptografados sob essa chave, o que significa que os dados podem se tornar irrecuperáveis. Você deve excluir uma chave somente quando tiver certeza de que não precisará mais usá-la e que nenhuma outra pessoa a utiliza. Se você não tiver certeza, considere interromper o uso da chave em vez de excluí-la. Você pode reativar uma chave desativada se precisar usá-la novamente mais tarde, mas não poderá recuperar uma chave de criptografia de AWS pagamento excluída, a menos que consiga reimportá-la de outra fonte.

Antes de excluir uma chave, você deve se certificar de que não precisa mais dela. AWS A criptografia de pagamento não armazena os resultados de operações criptográficas como CVV2 e não consegue determinar se uma chave é necessária para qualquer material criptográfico persistente.

AWS A criptografia de pagamento nunca exclui chaves pertencentes a AWS contas ativas, a menos que você as agende explicitamente para exclusão e o período de espera obrigatório expire.

No entanto, você pode optar por excluir uma chave de criptografia de AWS pagamento por um ou mais dos seguintes motivos:

- Para concluir o ciclo de vida de uma chave que você não precisa mais
- Para evitar a sobrecarga de gerenciamento associada à manutenção de chaves de criptografia de AWS pagamento não utilizadas

> **Note**
>
> Se você fechar ou excluir sua Conta da AWS, sua chave AWS de criptografia de pagamento ficará inacessível. Você não precisa programar a exclusão de sua chave de criptografia de AWS pagamento além do fechamento da conta.

AWS A criptografia de pagamento registra uma entrada em seu AWS CloudTrail registro quando você agenda a exclusão da chave de criptografia de AWS pagamento e quando a chave de criptografia de AWS pagamento é realmente excluída.

Ao usar a replicação de Multi-Region chaves, excluindo uma chave de criptografia de pagamento que seja uma chave de região primária (PRK), as chaves de região de réplica (RRK) também serão excluídas automaticamente. Um RRK não pode ser excluído como um PRK. Se quiser excluir um RRK, você precisará modificar as regiões de replicação do seu PRK.

Sobre o período de espera

Como a exclusão de uma chave é irreversível, a criptografia de AWS pagamento exige que você defina um período de espera entre 3 e 180 dias. O período de espera padrão é de sete dias.

No entanto, o período de espera real pode ser até 24 horas mais longo do que o programado. Para obter a data e a hora reais em que a chave AWS de criptografia de pagamento será excluída, use as GetKey operações. Certifique-se de anotar o fuso horário.

Durante o período de espera, o status da chave AWS de criptografia de pagamento e o estado da chave são Exclusão pendente.

> **Note**
>
> Uma chave AWS de criptografia de pagamento pendente de exclusão não pode ser usada em nenhuma operação criptográfica.

Após o término do período de espera, a criptografia de AWS pagamento exclui a chave de criptografia AWS de pagamento, seus aliases e todos os metadados relacionados à AWS criptografia de pagamento.

Use o período de espera para garantir que você não precise da chave AWS de criptografia de pagamento agora ou no futuro. Se você achar que precisa da chave durante o período de espera, basta cancelar a exclusão de chaves antes do término do período de espera. Após o término do período de espera, você não poderá cancelar a exclusão da chave e o serviço excluirá a chave.

Example

Neste exemplo, é solicitada a exclusão de uma chave. Além das informações básicas da chave, dois campos relevantes são que o estado da chave foi alterado para DELETE_PENDING e delete PendingTimestamp representa quando a chave está atualmente programada para ser excluída.

```
$ 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

Neste exemplo, uma exclusão pendente é cancelada. Depois de concluída com sucesso, uma chave não será mais excluída de acordo com a programação anterior. A resposta contém as principais informações básicas; além disso, dois campos relevantes foram alterados: `KeyState` e `deletePendingTimestamp`. `KeyState` é retornado para um valor de CREATE_COMPLETE, enquanto `DeletePendingTimestamp` é removido.

```
$ 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"
    }
}
```

Importando e exportando chaves

Você pode importar chaves AWS de criptografia de pagamento de outras soluções e exportá-las para outras soluções, como HSMs. Muitos clientes trocam chaves com provedores de serviços usando a funcionalidade de importação e exportação. Projetamos a criptografia de AWS pagamento para usar uma abordagem eletrônica moderna para o gerenciamento de chaves que ajuda você a manter a conformidade e os controles. Recomendamos usar o intercâmbio eletrônico de chaves baseado em padrões em vez de componentes de chave baseados em papel. Se você precisar continuar processando os principais componentes em papel até que todos os parceiros ofereçam suporte à troca eletrônica de chaves, você pode usarTroca de chaves físicas.

Pontos fortes mínimos das teclas e o efeito nas funções de importação e exportação

O PCI exige pontos fortes mínimos específicos para operações criptográficas, armazenamento e transmissão de chaves. Esses requisitos podem mudar quando os padrões PCI são revisados. As regras especificam que as chaves de empacotamento usadas para armazenamento ou transporte devem ser pelo menos tão fortes quanto a chave que está sendo protegida. Aplicamos esse requisito automaticamente durante a exportação e evitamos que as chaves sejam protegidas por chaves mais fracas, conforme mostrado na tabela a seguir.

A tabela a seguir mostra as combinações suportadas de chaves de empacotamento, chaves para proteção e métodos de proteção.

	Chave de embrulho											
Chave para proteger	TDES	TDES	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Observações
CHAVE TDES_2KE	TR-3	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	
CHAVE TDES_3	✗ Não supo	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	
AES_128	✗ Não	✗ Não	TR-3	TR-3	TR-3	✗ Não	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	

	Chave de embrulho											
Chave para proteger	TDES supor	TDES supor	AES_	AES_	AES_	RSA_ supor	RSA_	RSA_	ECC_	ECC_	ECC_	Observações
AES_192	✗ Não supor	✗ Não supor	✗ Não supor	TR-3	TR-3	✗ Não supor	✗ Não supor	✗ Não supor	✗ Não supor	ECDH	ECDH	
AES_256	✗ Não supor	✗ Não supor	✗ Não supor	✗ Não supor	TR-3	✗ Não supor	✗ Não supor	✗ Não supor	✗ Não supor	✗ Não supor	ECDH	

Para obter mais informações, consulte o Apêndice D - Tamanhos e pontos fortes mínimos e equivalentes de chaves para algoritmos aprovados nos padrões PCI HSM.

Troca de chave de criptografia de chave (KEK)

Recomendamos usar o X9.24 TR-34 padrão ANSI. Esse tipo de chave inicial pode ser chamado de chave de criptografia de chave (KEK), chave mestra de zona (ZMK) ou chave mestra de controle de zona (ZCMK). Se seus sistemas ou parceiros TR-34 ainda não oferecem suporte, você pode usar a RSA Wrap/Unwrap. Se suas necessidades incluírem a troca de AES-256 chaves, você pode usar o ??? ECDH.

> **Note**
>
> Para importar suas próprias chaves de teste ou sincronizar chaves com seus HSMs existentes, consulte o código de exemplo de criptografia AWS de pagamento em. GitHub

Troca de chaves de trabalho (WK)

Usamos padrões do setor (ANSI X9.24 TR 31-2018 e X9.143) para trocar chaves de trabalho. Isso requer que você já tenha trocado um KEK usando RSA Wrap TR-34, ECDH ou esquemas similares. Essa abordagem atende ao requisito de PIN do PCI para vincular criptograficamente o

material chave ao seu tipo e uso em todos os momentos. As chaves de trabalho incluem chaves de trabalho do adquirente, chaves de trabalho do emissor, BDK e IPEK.

Tópicos

- Importar chaves
- Exportar chaves
- Tópicos avançados

Importar chaves

Important

Os exemplos exigem a versão mais recente do AWS CLI V2. Antes de começar, certifique-se de ter atualizado para a versão https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html mais recente.

Sumário

- Introdução à importação de chaves
- Importar chaves simétricas
  - Importar chaves usando técnicas assimétricas () TR-34
  - Importar chaves usando técnicas assimétricas (ECDH)
  - Importe chaves usando técnicas assimétricas (RSA Unwrap)
  - Importe chaves simétricas usando uma chave de troca de chaves pré-estabelecida () TR-31
- Importação de chaves públicas assimétricas (RSA, ECC)
  - Importar chaves públicas RSA
  - Importação de chaves públicas ECC

Introdução à importação de chaves

Note

Ao importar chaves usando TR-31 ou blocos de chaves X9.143, a criptografia de AWS pagamento normalmente retém (mas não utiliza) TR-34 cabeçalhos opcionais. O cabeçalho

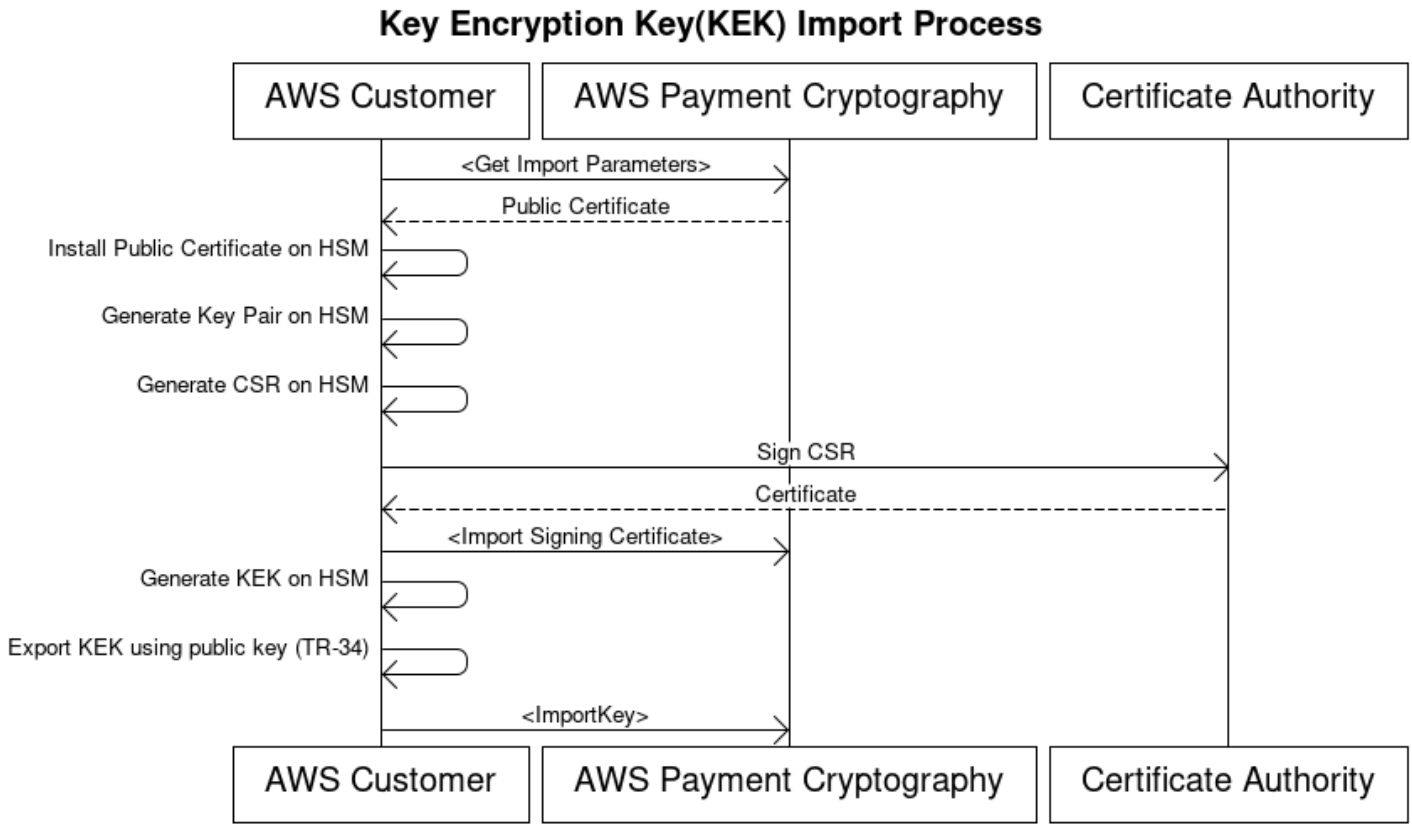
HM (tipo de hash HMAC) é usado durante operações criptográficas. O cabeçalho KP (KCV da chave de empacotamento) é específico para o processo de importação e não é retido.

Ao trocar chaves com uma contraparte, normalmente é para trocar primeiro uma chave de troca de chaves (KEK). Essa chave será então usada para proteger as chaves subsequentes. Usando formatos eletrônicos, o KEK pode ser trocado usando técnicas assimétricas TR-34, como ECDH ou RSA wrap. As chaves subsequentes serão trocadas usando uma troca de chaves simétrica, como. TR-31 Este KEK terá vida longa e só poderá ser atualizado a cada poucos anos com base na política e no período criptográfico definido.

Se apenas uma ou duas chaves estiverem sendo trocadas, você também poderá optar por usar técnicas assimétricas para trocar diretamente essa chave, como um BDK. AWS A criptografia de pagamento suporta os dois métodos de troca de chaves.

Importar chaves simétricas

Importar chaves usando técnicas assimétricas () TR-34



TR-34 usa criptografia assimétrica RSA para criptografar e assinar chaves simétricas para troca. Isso garante a confidencialidade (criptografia) e a integridade (assinatura) da chave encapsulada.

Para importar suas próprias chaves, confira o projeto de amostra AWS de criptografia de pagamento em GitHub. Para obter instruções sobre como usar import/export chaves de outras plataformas, o código de exemplo está disponível GitHub ou consulte o guia do usuário dessas plataformas.

1. Chame o comando Inicializar importação

   Chame `get-parameters-for-import` para inicializar o processo de importação. Essa API gera um par de chaves para importações de chaves, assina a chave e retorna o certificado e a raiz do certificado. Criptografe a chave a ser exportada usando essa chave. Na TR-34 terminologia, isso é conhecido como KRD Cert. Esses certificados são codificados em base64, têm vida curta e são destinados somente para essa finalidade. Salve o `ImportToken` valor.

   ```
   $ 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. Instalar o certificado público no sistema de origem da chave

   Com a maioria dos HSMs, você precisa instalar, carregar ou confiar no certificado público gerado na etapa 1 para exportar chaves usando-o. Isso pode incluir toda a cadeia de certificados ou apenas o certificado raiz da etapa 1, dependendo do HSM.

3. Gere um par de chaves no sistema de origem e forneça a cadeia de certificados para AWS Payment Cryptography

   Para garantir a integridade da carga transmitida, a parte remetente (Key Distribution Host ou KDH) a assina. Gere uma chave pública para essa finalidade e crie um certificado de chave pública (X509) para devolver à criptografia de AWS pagamento.

Ao transferir chaves de um HSM, crie um par de chaves nesse HSM. O HSM, um terceiro ou um serviço como esse CA privada da AWS pode gerar o certificado.

Carregue o certificado raiz na criptografia de AWS pagamento usando o `importKey` comando com KeyMaterialType of `RootCertificatePublicKey` e KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Para certificados intermediários, use o `importKey` comando com KeyMaterialType of `TrustedCertificatePublicKey` e KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Repita esse processo para vários certificados intermediários. Use o `KeyArn` último certificado importado na cadeia como entrada para os comandos de importação subsequentes.

> **Note**
>
> Não importe o certificado foliar. Forneça-o diretamente durante o comando de importação.

4. Exportar chave do sistema de origem

   Muitos HSMs e sistemas relacionados suportam a exportação de chaves usando a TR-34 norma. Especifique a chave pública da etapa 1 como o certificado KRD (criptografia) e a chave da etapa 3 como o certificado KDH (assinatura). Para importar para a criptografia de AWS pagamento, especifique o formato como formato de duas passagens TR-34.2012 não CMS, que também pode ser chamado de formato Diebold. TR-34

5. Chave de importação de chamadas

   Chame a API importKey com um KeyMaterialType de. `TR34_KEY_BLOCK` Use o keyARN da última CA importada na etapa 3 para`certificate-authority-public-key-identifier`, o material de chave encapsulado da etapa 4 para `key-material` e o certificado de folha da etapa 3 para. `signing-key-certificate` Inclua o token de importação da etapa 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", \
```

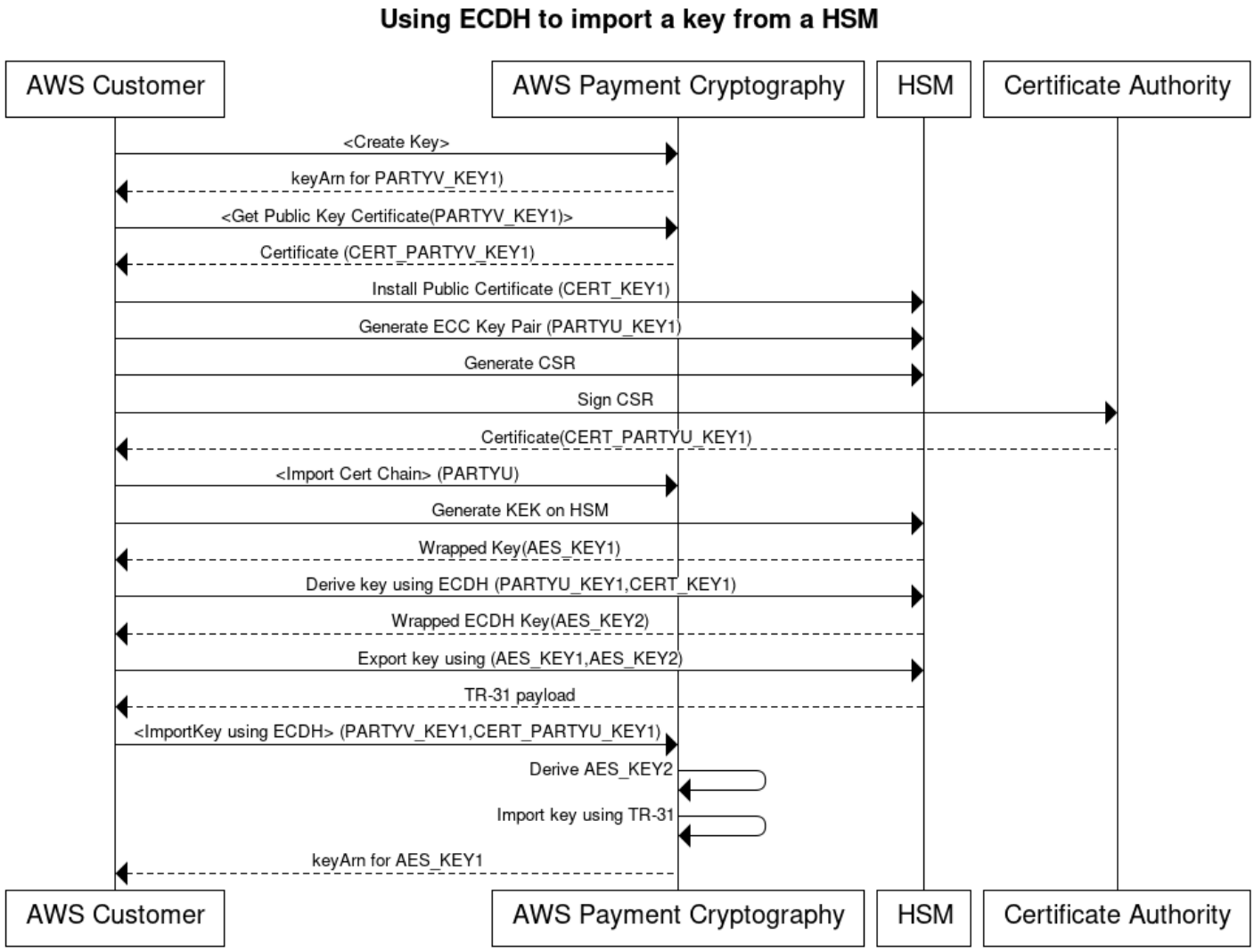
```
    "SigningKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJ...", \
    "WrappedKeyBlock":
 "308205A106092A864886F70D010702A08205923082058E020101310D300B0609608648016503040201308203.
 \
    }'
```

```
{
  "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. Use a chave importada para operações criptográficas ou importação subsequente

   Se a chave KeyUsage importada for TR31_K0_KEY_ENCRYPTION_KEY, você poderá usar essa chave para futuras importações de chaves. TR-31 Para outros tipos de chave (como TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), você pode usar a chave diretamente para operações criptográficas.

Importar chaves usando técnicas assimétricas (ECDH)



A curva elíptica Diffie-Hellman (ECDH) usa criptografia assimétrica ECC para estabelecer uma chave compartilhada entre duas partes sem exigir chaves pré-trocadas. As chaves ECDH são efêmeras, portanto, a criptografia de AWS pagamento não as armazena. Nesse processo, uma única vez KBPK/KEK é derivada usando ECDH. Essa chave derivada é usada imediatamente para agrupar a chave real que você deseja transferir, que pode ser outra KBPK, uma chave IPEK ou outros tipos de chave.

Ao importar, o sistema de envio é comumente conhecido como Parte U (Iniciador) e a Criptografia de AWS Pagamento é conhecida como Parte V (Respondente).

> **Note**
>
> Embora o ECDH possa ser usado para trocar qualquer tipo de chave simétrica, é a única abordagem que pode transferir chaves com segurança. AES-256

1. Gerar par de chaves ECC

   Ligue `create-key` para criar um par de chaves ECC para esse processo. Essa API gera um par de chaves para importações ou exportações de chaves. Na criação, especifique que tipo de chaves podem ser derivadas usando essa chave ECC. Ao usar o ECDH para trocar (embrulhar) outras chaves, use um valor de. `TR31_K1_KEY_BLOCK_PROTECTION_KEY`

   > **Note**
   >
   > Embora o ECDH de baixo nível gere uma chave derivada que pode ser usada para qualquer finalidade, a criptografia de AWS pagamento limita a reutilização acidental de uma chave para várias finalidades, permitindo que uma chave seja usada somente para um único tipo de chave derivada.

```
$ 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. Obtenha o certificado de chave pública

   Ligue `get-public-key-certificate` para receber a chave pública como um X.509 certificado assinado pela CA da sua conta que é específico para criptografia AWS de pagamento em uma região específica.

   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. Instale o certificado público no sistema da contraparte (Parte U)

   Com muitos HSMs, você precisa instalar, carregar ou confiar no certificado público gerado na etapa 1 para exportar chaves usando-o. Isso pode incluir toda a cadeia de certificados ou apenas o certificado raiz da etapa 1, dependendo do HSM. Consulte a documentação do HSM para obter mais informações.

4. Gere um par de chaves ECC no sistema de origem e forneça uma cadeia de certificados para AWS Payment Cryptography

   No ECDH, cada parte gera um par de chaves e concorda com uma chave comum. Para que a criptografia de AWS pagamento obtenha a chave, ela precisa da chave pública da contraparte no formato de chave X.509 pública.

   Ao transferir chaves de um HSM, crie um par de chaves nesse HSM. Para HSMs que suportam blocos de chaves, o cabeçalho da chave será semelhante a. `D0144K3EX00E0000` Ao criar o certificado, você geralmente gera uma CSR no HSM e, em seguida, o HSM, um terceiro ou um serviço como esse CA privada da AWS pode gerar o certificado.

   Carregue o certificado raiz na criptografia de AWS pagamento usando o `importKey` comando com KeyMaterialType of `RootCertificatePublicKey` e KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

   Para certificados intermediários, use o `importKey` comando com KeyMaterialType of `TrustedCertificatePublicKey` e KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Repita esse processo para vários certificados intermediários. Use o `KeyArn` último certificado importado na cadeia como entrada para os comandos de importação subsequentes.

   > **Note**
   >
   > Não importe o certificado foliar. Forneça-o diretamente durante o comando de importação.

5. Derive uma chave única usando ECDH no Party U HSM

   Muitos HSMs e sistemas relacionados suportam o estabelecimento de chaves usando ECDH. Especifique a chave pública da etapa 1 como a chave pública e a chave da etapa 3 como a chave privada. Para opções permitidas, como métodos de derivação, consulte o guia da API.

   > **Note**
   >
   > Os parâmetros de derivação, como o tipo de hash, devem corresponder exatamente nos dois lados. Caso contrário, você gerará uma chave diferente.

6. Exportar chave do sistema de origem

Por fim, exporte a chave que você deseja transportar para a criptografia AWS de pagamento usando TR-31 comandos padrão. Especifique a chave derivada do ECDH como KBPK. A chave a ser exportada pode ser qualquer chave TDES ou AES sujeita a combinações TR-31 válidas, desde que a chave de empacotamento seja pelo menos tão forte quanto a chave a ser exportada.

7. Chave de importação de chamadas

   Chame a `import-key` API com um KeyMaterialType de`DiffieHellmanTr31KeyBlock`. Use o keyARN da última CA importada na etapa 3 para`certificate-authority-public-key-identifier`, o material de chave encapsulado da etapa 4 para `key-material` e o certificado de folha da etapa 3 para. `public-key-certificate` Inclua a chave privada ARN da etapa 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. Use a chave importada para operações criptográficas ou importação subsequente

   Se a chave KeyUsage importada for TR31_K0_KEY_ENCRYPTION_KEY, você poderá usar essa chave para futuras importações de chaves. TR-31 Para outros tipos de chave (como TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), você pode usar a chave diretamente para operações criptográficas.

Importe chaves usando técnicas assimétricas (RSA Unwrap)

Visão geral: a criptografia AWS de pagamento oferece suporte ao RSA wrap/unwrap para troca de chaves quando TR-34 não é viável. Por exemplo TR-34, essa técnica usa criptografia assimétrica RSA para criptografar chaves simétricas para troca. No entanto, ao contrário disso TR-34, esse método não faz com que a parte remetente assine a carga. Além disso, essa técnica de encapsulamento RSA não mantém a integridade dos metadados da chave durante a transferência porque não inclui blocos de chaves.

> **Note**
>
> Você pode usar o RSA wrap para importar ou exportar TDES e AES-128 chaves.

1. Chame o comando Inicializar importação

   Ligue get-parameters-for-import para inicializar o processo de importação com um `KeyMaterialType` of`KEY_CRYPTOGRAM`. Use `RSA_2048` para o `WrappingKeyAlgorithm` ao trocar chaves TDES. Use `RSA_3072` ou `RSA_4096` ao trocar TDES ou AES-128 chaves. Essa API gera um par de chaves para importações de chaves, assina a chave usando uma raiz de certificado e retorna o certificado e a raiz do certificado. Criptografe a chave a ser exportada usando essa chave. Esses certificados têm vida curta e se destinam apenas a essa finalidade.

   ```
   $ 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. Instalar o certificado público no sistema de origem da chave

   Com muitos HSMs, você precisa instalar, carregar ou confiar no certificado público (and/orsua raiz) gerado na etapa 1 para exportar chaves usando-o.

3. Exportar chave do sistema de origem

   Muitos HSMs e sistemas relacionados oferecem suporte à exportação de chaves usando o RSA wrap. Especifique a chave pública da etapa 1 como o certificado de criptografia (`WrappingKeyCertificate`). Se você precisar da cadeia de confiança, use a `WrappingKeyCertificateChain` partir da etapa 1. Ao exportar a chave do seu HSM, especifique o formato como RSA, com Padding Mode = PKCS #1 v2.2 OAEP (com SHA 256 ou SHA 512).

4. Chame chave de importação

   Chame a import-key API com um `KeyMaterialType` de`KeyMaterial`. Você precisa `ImportToken` da etapa 1 e do `key-material` (material da chave embrulhada) da etapa 3. Forneça os principais parâmetros (como uso de chaves) porque o RSA wrap não usa blocos de chaves.

```
$ 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
```

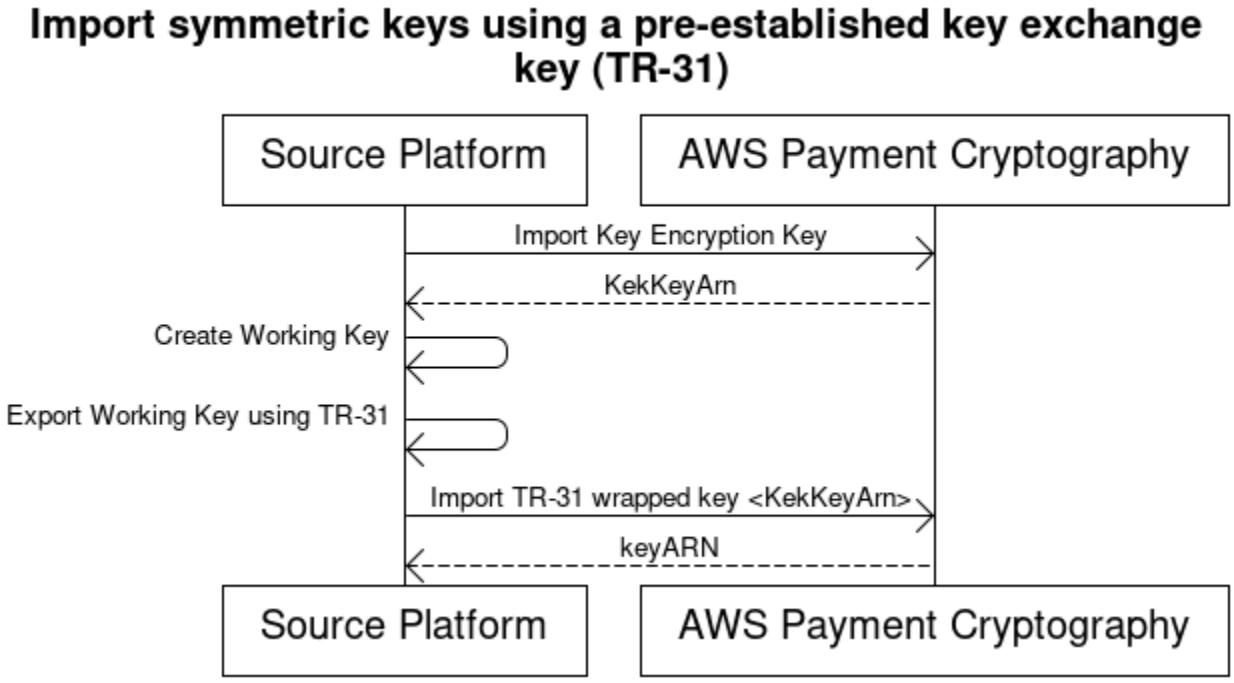
```
{
  "Key": {
    "KeyOrigin": "EXTERNAL",
    "Exportable": true,
    "KeyCheckValue": "DA1ACF",
    "UsageStartTimestamp": 1697643478.92,
    "Enabled": true,
```

```
    "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. Use a chave importada para operações criptográficas ou importação subsequente

   Se o importado `KeyUsage` foi `TR31_K0_KEY_ENCRYPTION_KEY` ou`TR31_K1_KEY_BLOCK_PROTECTION_KEY`, você pode usar essa chave para importações de chaves subsequentes usando TR-31. Se o tipo de chave for de qualquer outro tipo (como`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`), você poderá usar a chave diretamente para operações criptográficas.

Importe chaves simétricas usando uma chave de troca de chaves pré-estabelecida () TR-31



Ao trocar várias chaves ou oferecer suporte à rotação de chaves, os parceiros normalmente trocam primeiro uma chave inicial de criptografia de chave (KEK). Você pode trocar KEK por criptografia AWS de pagamento, usando técnicas como TR-34 ou. Troca de chaves físicas

Depois de estabelecer uma KEK, você pode usá-la para transportar as chaves subsequentes (incluindo outras KEKs). AWS A criptografia de pagamento suporta essa troca de chaves usando o ANSI TR-31, que é amplamente usado e suportado por fornecedores de HSM.

1. Importar chave de criptografia de chave (KEK)

   Verifique se você já importou seu KEK e tem o keyArn (ou keyAlias) disponível.

2. Criar uma chave na plataforma de origem

   Se a chave não existir, crie-a na plataforma de origem. Como alternativa, você pode criar a chave na Criptografia AWS de pagamento e usar o export comando.

3. Exportar chave da plataforma de origem

   Ao exportar, especifique o formato de exportação como TR-31. A plataforma de origem solicitará a chave a ser exportada e a chave de criptografia de chave a ser usada.

4. Importar para AWS Payment Cryptography

Ao chamar o import-key comando, use o keyARN (ou alias) da sua chave de criptografia de chave para. `WrappingKeyIdentifier` Use a saída da plataforma de origem para`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"
  }
}
```

Importação de chaves públicas assimétricas (RSA, ECC)

Todos os certificados importados devem ser pelo menos tão fortes quanto o certificado emissor (antecessor) na cadeia. Isso significa que uma CA RSA_2048 só pode ser usada para proteger um certificado de folha RSA_2048 e um certificado ECC deve ser protegido por outro certificado ECC de força equivalente. Um certificado ECC P384 só pode ser emitido por uma CA P384 ou P521. Todos os certificados não devem estar expirados no momento da importação.

Importar chaves públicas RSA

AWS A criptografia de pagamento suporta a importação de chaves RSA públicas como certificados. X.509 Para importar um certificado, primeiro importe seu certificado raiz. Todos os certificados não devem estar expirados no momento da importação. O certificado deve estar no formato PEM e ser codificado em base64.

1. Importar certificado raiz para AWS Payment Cryptography

   Use o comando a seguir para importar o certificado raiz:

Example

2. Importar certificado de chave pública para AWS Payment Cryptography

   Agora, você pode importar uma chave pública. Como TR-34 o ECDH depende da aprovação do certificado Leaf em tempo de execução, essa opção só é usada ao criptografar dados usando uma chave pública de outro sistema. KeyUsage será definido como 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"
  }
}
```

Importação de chaves públicas ECC

AWS A criptografia de pagamento suporta a importação de chaves ECC públicas como certificados. X.509 Para importar um certificado, primeiro importe seu certificado CA raiz e quaisquer certificados intermediários. Todos os certificados não devem estar expirados no momento da importação. O certificado deve estar no formato PEM e ser codificado em base64.

1. Importar certificado raiz ECC para AWS Payment Cryptography

   Use o comando a seguir para importar o certificado raiz:

Example

2. Importar certificado intermediário para AWS Payment Cryptography

   Use o comando a seguir para importar um certificado intermediário:

Example

3. Importar certificado de chave pública (folha) para AWS Payment Cryptography

   Embora você possa importar um certificado ECC de folha, atualmente não há nenhuma função definida na Criptografia de AWS Pagamento para ele além do armazenamento. Isso ocorre porque, ao usar as funções ECDH, o certificado leaf é passado em tempo de execução.

Exportar chaves

Sumário

- Exportar chaves simétricas
  - Exportar chaves usando técnicas assimétricas () TR-34
  - Exporte chaves usando técnicas assimétricas (ECDH)
  - Exporte chaves usando técnicas assimétricas (RSA Wrap)
  - Exporte chaves simétricas usando uma chave de troca de chaves preestabelecida () TR-31
- Exportar chaves iniciais DUKPT () IPEK/IK
- Especifique cabeçalhos de blocos de chaves para exportação
  - Cabeçalhos comuns
- Exportar chaves assimétricas (RSA)

Exportar chaves simétricas

> **Important**
>
> Certifique-se de ter a versão mais recente do AWS CLI antes de começar. Para atualizar, consulte Instalando AWS CLI o.

Exportar chaves usando técnicas assimétricas () TR-34

TR-34 usa criptografia assimétrica RSA para criptografar e assinar chaves simétricas para troca. A criptografia protege a confidencialidade, enquanto a assinatura garante a integridade. Quando você exporta chaves, a criptografia de AWS pagamento atua como o host de distribuição de chaves (KDH) e seu sistema de destino se torna o dispositivo receptor de chaves (KRD).

> **Note**
>
> Se seu HSM oferece suporte TR-34 à exportação, mas não à TR-34 importação, recomendamos que você primeiro estabeleça uma KEK compartilhada entre o HSM e a criptografia de AWS pagamento usando. TR-34 Você pode então usar TR-31 para transferir suas chaves restantes.

1. Inicializar o processo de exportação

   Execute get-parameters-for-export para gerar um par de chaves para exportações de chaves. Usamos esse par de chaves para assinar a TR-34 carga útil. Na TR-34 terminologia, esse é o certificado de assinatura KDH. Os certificados têm vida curta e são válidos somente pelo período especificado em`ParametersValidUntilTimestamp`.

   > **Note**
   >
   > Todos os certificados estão na codificação 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. Importe o AWS Certificado de criptografia de pagamento para seu sistema de recebimento

   Importe a cadeia de certificados da etapa 1 para o sistema de recebimento.

3. Configure os certificados do seu sistema de recebimento

Para proteger a carga transmitida, a parte remetente (KDH) a criptografa. Seu sistema receptor (normalmente seu HSM ou o HSM do seu parceiro) precisa gerar uma chave pública e criar um certificado de chave X.509 pública. Você pode usar CA privada da AWS para gerar certificados, mas você pode usar qualquer autoridade de certificação.

Depois de obter o certificado, importe o certificado raiz para a Criptografia de AWS pagamento usando o ImportKey comando. Defina `KeyMaterialType` como `RootCertificatePublicKey` e `KeyUsageType` como `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Usamos `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` como o `KeyUsageType` porque essa é a chave raiz que assina o certificado foliar. Você não precisa importar certificados Leaf para a criptografia AWS de pagamento — você pode passá-los on-line.

> **Note**
>
> Se você importou anteriormente o certificado raiz, ignore esta etapa. Para certificados intermediários, use`TrustedCertificatePublicKey`.

4. Exporte sua chave

   Chame a ExportKey API com `KeyMaterialType` set to`TR34_KEY_BLOCK`. Você precisa fornecer:

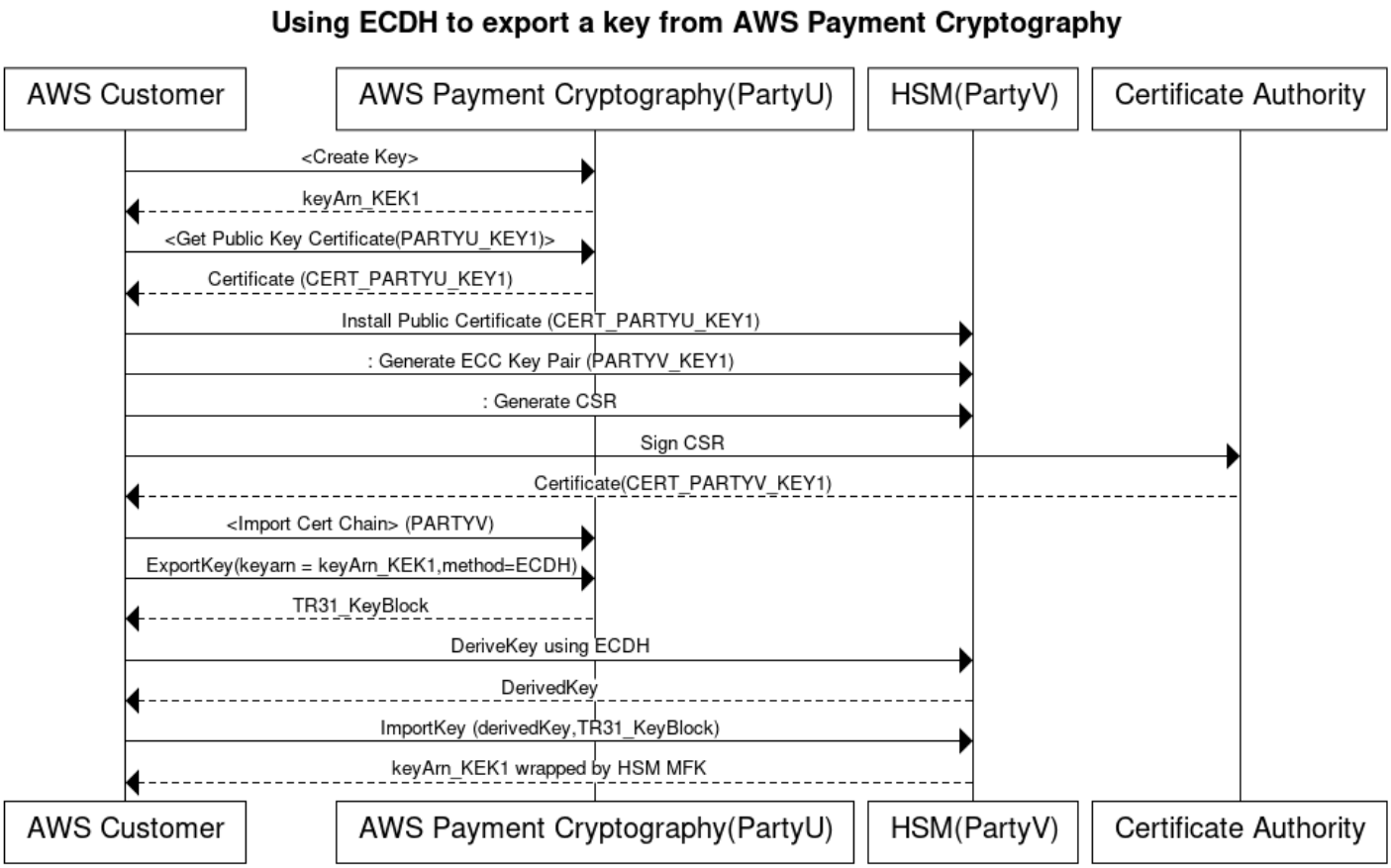
   - O keyARN da CA raiz da etapa 3 como o `CertificateAuthorityPublicKeyIdentifier`
   - O certificado foliar da etapa 3 como `WrappingKeyCertificate`
   - O keyArn (ou alias) da chave que você deseja exportar como `--export-key-identifier`
   - O token de exportação da etapa 1

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"
  }
}
```

Exporte chaves usando técnicas assimétricas (ECDH)



A curva elíptica Diffie-Hellman (ECDH) usa criptografia assimétrica ECC para estabelecer uma chave compartilhada entre duas partes sem exigir chaves pré-trocadas. As chaves ECDH são efêmeras, portanto, a criptografia de AWS pagamento não as armazena. Nesse processo, uma única vez KBPK/KEK é derivada usando ECDH. Essa chave derivada é usada imediatamente para agrupar a chave que você deseja transferir, que pode ser outra chave KBPK, BDK, IPEK ou outros tipos de chave.

Ao exportar, a Criptografia de AWS Pagamento é chamada de Parte U (Iniciador) e o sistema de recebimento é conhecido como Parte V (Respondente).

> **Note**
>
> O ECDH pode ser usado para trocar qualquer tipo de chave simétrica, mas é a única abordagem que pode ser usada para transferir AES-256 chaves se uma KEK ainda não estiver estabelecida.

1. Gerar par de chaves ECC

   Ligue `create-key` para criar um par de chaves ECC para esse processo. Essa API gera um par de chaves para importações ou exportações de chaves. Na criação, especifique que tipo de chaves podem ser derivadas usando essa chave ECC. Ao usar o ECDH para trocar (embrulhar) outras chaves, use um valor de. `TR31_K1_KEY_BLOCK_PROTECTION_KEY`

   > **Note**
   >
   > Embora o ECDH de baixo nível gere uma chave derivada que pode ser usada para qualquer finalidade, a criptografia de AWS pagamento limita a reutilização acidental de uma chave para várias finalidades, permitindo que uma chave seja usada somente para um único tipo de chave derivada.

```
$ 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. Obtenha o certificado de chave pública

   Ligue `get-public-key-certificate` para receber a chave pública como um X.509 certificado assinado pela CA da sua conta que é específico para criptografia AWS de pagamento em uma região específica.

   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. Instalar certificado público no sistema da contraparte (Parte V)

   Com muitos HSMs, você precisa instalar, carregar ou confiar no certificado público gerado na etapa 1 para estabelecer as chaves. Isso pode incluir toda a cadeia de certificados ou apenas o certificado raiz, dependendo do HSM. Consulte a documentação do HSM para obter instruções específicas.

4. Gere um par de chaves ECC no sistema de origem e forneça uma cadeia de certificados para AWS Payment Cryptography

   No ECDH, cada parte gera um par de chaves e concorda com uma chave comum. Para que a criptografia de AWS pagamento obtenha a chave, ela precisa da chave pública da contraparte no formato de chave X.509 pública.

Ao transferir chaves de um HSM, crie um par de chaves nesse HSM. Para HSMs que suportam blocos de chaves, o cabeçalho da chave será semelhante a. `D0144K3EX00E0000` Ao criar o certificado, você geralmente gera uma CSR no HSM e, em seguida, o HSM, um terceiro ou um serviço como esse CA privada da AWS pode gerar o certificado.

Carregue o certificado raiz na criptografia de AWS pagamento usando o `importKey` comando com KeyMaterialType of `RootCertificatePublicKey` e KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Para certificados intermediários, use o `importKey` comando com KeyMaterialType of `TrustedCertificatePublicKey` e KeyUsageType of`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Repita esse processo para vários certificados intermediários. Use o `KeyArn` último certificado importado na cadeia como entrada para os comandos de exportação subsequentes.

> **Note**
>
> Não importe o certificado foliar. Forneça-o diretamente durante o comando de exportação.

5. Derive a chave e exporte a chave de AWS Payment Cryptography

Ao exportar, o serviço deriva uma chave usando ECDH e a usa imediatamente como KBPK para encapsular a chave a ser usada para exportar. TR-31 A chave a ser exportada pode ser qualquer chave TDES ou AES sujeita a combinações TR-31 válidas, desde que a chave de empacotamento seja pelo menos tão forte quanto a chave a ser exportada.

```
$ 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. Derive uma chave única usando ECDH no Party V HSM

   Muitos HSMs e sistemas relacionados suportam o estabelecimento de chaves usando ECDH. Especifique a chave pública da etapa 1 como a chave pública e a chave da etapa 3 como a chave privada. Para opções permitidas, como métodos de derivação, consulte o guia da API.

   > **Note**
   >
   > Os parâmetros de derivação, como o tipo de hash, devem corresponder exatamente nos dois lados. Caso contrário, você gerará uma chave diferente.

7. Importar chave para o sistema de destino

   Por fim, importe a chave da criptografia de AWS pagamento usando TR-31 comandos padrão. Especifique a chave derivada do ECDH como KBPK e use o bloco de TR-31 chaves que foi exportado anteriormente da criptografia de pagamento. AWS

Exporte chaves usando técnicas assimétricas (RSA Wrap)

Quando TR-34 não estiver disponível, você pode usar o RSA wrap/unwrap para troca de chaves. Por exemplo TR-34, esse método usa criptografia assimétrica RSA para criptografar chaves simétricas. No entanto, o RSA wrap não inclui:

- Assinatura de carga útil pela parte remetente
- Blocos principais que mantêm a integridade dos metadados principais durante o transporte

> **Note**
>
> Você pode usar o RSA wrap para exportar TDES e AES-128 chaves.

1. Crie uma chave RSA e um certificado em seu sistema receptor

   Crie ou identifique uma chave RSA para receber a chave encapsulada. Exigimos que as chaves estejam em formato X.509 de certificado. Certifique-se de que o certificado esteja assinado por um certificado raiz que você possa importar para a criptografia AWS de pagamento.
2. Importe o certificado público raiz para AWS Payment Cryptography

   Use import-key com a `--key-material` opção de importar o certificado

```
$ 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. Exporte sua chave

   Peça à AWS Payment Cryptography que exporte sua chave usando seu certificado Leaf. Você precisa especificar:

   - O ARN do certificado raiz que você importou na etapa 2
   - O certificado de folha para exportação
   - A chave simétrica para exportar

   A saída é uma versão binária encapsulada (criptografada) com codificação hexadecimal da sua chave simétrica.

Example Exemplo — Exportar uma chave

```
$ 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. Importe a chave para seu sistema de recebimento

   Muitos HSMs e sistemas relacionados oferecem suporte à importação de chaves usando o RSA unwrap (incluindo AWS criptografia de pagamento). Ao importar, especifique:

   - A chave pública da etapa 1 como certificado de criptografia
   - O formato como RSA
   - Modo de preenchimento como PKCS #1 v2.2 OAEP (com SHA 256)

> **Note**
>
> Produzimos a chave encapsulada no formato hexBinary. Talvez seja necessário converter o formato se o sistema exigir uma representação binária diferente, como base64.

Exporte chaves simétricas usando uma chave de troca de chaves preestabelecida () TR-31

Ao trocar várias chaves ou oferecer suporte à rotação de chaves, os parceiros normalmente trocam primeiro uma chave inicial de criptografia de chave (KEK). Você pode trocar KEK por criptografia AWS de pagamento, usando técnicas como TR-34 ou. Troca de chaves físicas Depois de estabelecer uma KEK, você pode usá-la para transportar as chaves subsequentes, incluindo outras KEKs. Apoiamos essa troca de chaves usando o ANSI TR-31, que é amplamente suportado pelos fornecedores de HSM.

1. Configurar sua chave de criptografia de chave (KEK)

   Verifique se você já trocou seu KEK e tem o keyARN (ou keyAlias) disponível.

2. Crie sua chave em AWS Payment Cryptography

   Crie sua chave se ela ainda não existir. Como alternativa, você pode criar a chave em seu outro sistema e usar o comando de importação.

3. Exporte sua chave do AWS Payment Cryptography

   Ao exportar em TR-31 formato, especifique a chave que você deseja exportar e a chave de empacotamento a ser usada.

Example Exemplo — Exportação de uma chave usando o bloco de teclas 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. Importe a chave para o seu sistema

   Use a implementação da chave de importação do seu sistema para importar a chave.

Exportar chaves iniciais DUKPT () IPEK/IK

Ao usar o DUKPT, você pode gerar uma única chave de derivação básica (BDK) para uma frota de terminais. Os terminais não têm acesso direto ao BDK. Em vez disso, cada terminal recebe uma chave de terminal inicial exclusiva, conhecida como IPEK ou Chave Inicial (IK). Cada IPEK é derivado do BDK usando um KSN (Key Serial Number) exclusivo.

A estrutura da KSN varia de acordo com o tipo de criptografia:

- Para TDES: O KSN de 10 bytes inclui:
  - 24 bits para o ID do conjunto de chaves
  - 19 bits para o ID do terminal
  - 21 bits para o contador de transações
- Para AES: O KSN de 12 bytes inclui:
  - 32 bits para o ID BDK

- 32 bits para o identificador de derivação (ID)
- 32 bits para o contador de transações

Fornecemos um mecanismo para gerar e exportar essas chaves iniciais. Você pode exportar as chaves geradas usando os métodos TR-31 TR-34,, ou RSA wrap. Observe que as chaves IPEK não persistem e não podem ser usadas para operações subsequentes na AWS criptografia de pagamento.

Não aplicamos a divisão entre as duas primeiras partes do KSN. Se quiser armazenar o identificador de derivação com o BDK, você pode usar tags. AWS

> **Note**
>
> A parte contrária do KSN (32 bits para AES DUKPT) não é usada para derivação. IPEK/IK Por exemplo, as entradas de 12345678901234560001 e 12345678901234569999 gerarão o mesmo 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"
}
}
```

Especifique cabeçalhos de blocos de chaves para exportação

Você pode modificar ou acrescentar informações do bloco de chaves ao exportar em TR-31 ASC ou formatos. TR-34 A tabela a seguir descreve o formato do bloco de TR-31 chaves e quais elementos você pode modificar durante a exportação.

Atributo do bloco chave	Finalidade	Você pode modificar durante a exportação?	Observações
ID da versão	Define o método usado para proteger o material da chave. O padrão inclui: • Versões A e C (variante chave - obsoleta) • Versão B (derivação usando TDES) • Versão D (derivação de chave usando AES)	Não	Usamos a versão B para chaves de encapsulamento TDES e a versão D para chaves de encapsulamento AES. Oferecemos suporte às versões A e C somente para operações de importação.
Comprimento do bloco de teclas	Especifica o tamanho da mensagem restante	Não	Calculamos esse valor automaticamente. O comprimento pode parecer incorreto antes de descriptografar a carga, pois podemos adicionar preenchimento de chave conforme exigido pela especificação.

Atributo do bloco chave	Finalidade	Você pode modificar durante a exportação?	Observações
Uso da chave	Define as finalidades permitidas para a chave, como: • C0 (Verificação do cartão) • B0 (chave de derivação básica)	Não	
Algoritmo	Especifica o algoritmo da chave subjacente. Nós apoiamos: • (MARÉS) • H (HMAC) • A (AES)	Não	Exportamos esse valor no estado em que se encontra.
Uso da chave	Define as operações permitidas, como: • Gerar e verificar (C) • Encrypt/Decrypt/Wrap/Unwrap (B)	Sim*	
Versão chave	Indica o número da versão da chave replacement/rotation. O padrão é 00 se não for especificado.	Sim - Pode acrescentar	

Atributo do bloco chave	Finalidade	Você pode modificar durante a exportação?	Observações
Exportabilidade chave	Controla se a chave pode ser exportada: • N - Sem exportabilidade • E - Exportar de acordo com X9.24 (blocos principais) • S - Exportar em formatos de bloco de chave ou de bloco não chave	Sim*	
Blocos de chaves opcionais	Sim - Pode acrescentar	Os blocos de chaves opcionais são name/value pares vinculados criptograficamente à chave. Por exemplo, KeySet ID para chaves DUKPT. Calculamos automaticamente o número de blocos, o comprimento de cada bloco e o bloco de preenchimento (PB) com base na entrada do seu name/value par.	

*Ao modificar valores, seu novo valor deve ser mais restritivo do que o valor atual na AWS Criptografia de pagamento. Por exemplo:

- Se o modo de uso da chave atual for Generate=True, Verify=True, você pode alterá-lo para Generate=True, Verify=False
- Se a chave já estiver definida como não exportável, você não poderá alterá-la para exportável

Quando você exporta chaves, aplicamos automaticamente os valores atuais da chave que está sendo exportada. No entanto, talvez você queira modificar ou acrescentar esses valores antes de enviar para o sistema receptor. Aqui estão alguns cenários comuns:

- Ao exportar uma chave para um terminal de pagamento, defina sua exportabilidade como `Not Exportable` porque os terminais normalmente importam apenas chaves e não devem exportá-las.
- Quando você precisar passar os metadados da chave associada para o sistema receptor, use cabeçalhos TR-31 opcionais para vincular criptograficamente os metadados à chave em vez de criar uma carga personalizada.
- Defina a versão da chave usando o `KeyVersion` campo para rastrear a rotação da chave.

TR-31/X9.143 define cabeçalhos comuns, mas você pode usar outros cabeçalhos, desde que eles atendam aos parâmetros AWS de criptografia de pagamento e seu sistema de recebimento possa aceitá-los. Para obter mais informações sobre cabeçalhos de blocos de chaves durante a exportação, consulte Cabeçalhos de blocos de chaves no Guia de API.

Aqui está um exemplo de exportação de uma chave BDK (por exemplo, para um KIF) com essas especificações:

- Versão chave: 02
- KeyExportability: NÃO EXPORTÁVEL
- KeySetID: 00ABCDEFAB (00 indica a chave TDES, ABCDEFABCD é a chave inicial)

Como não especificamos os modos de uso da chave, essa chave herda o modo de uso de arn:aws:payment-cryptography:us-east- 2:111122223333: (= true). key/5rplquuwozodpwsp DeriveKey

> **Note**
>
> Mesmo quando você define a exportabilidade como Não exportável neste exemplo, o KIF ainda pode:

- Chaves derivadas, como as IPEK/IK usadas no DUKPT
- Exporte essas chaves derivadas para instalar em dispositivos

Isso é especificamente permitido pelos padrões.

```
$ 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"
    }
}
```

Cabeçalhos comuns

X9.143 define certos cabeçalhos para casos de uso comuns. Com exceção do cabeçalho HM (HMAC Hash), a criptografia de AWS pagamento não analisa nem utiliza esses cabeçalhos.

Nome do cabeçalho	Finalidade	Validação típica	Observações
BI	Identificador de chave de derivação básica para DUKPT	2 caracteres hexadecimais (00 para TDES, 11 para AES) e 10 caracteres hexadecimais para TDES KSI ou 8 caracteres hexadecimais para BDK ID (AES DUKPT).	Contém o (ID BDK, para AES DUKPT) ou o Key Set Identifier (KSI, para TDES DUKPT). Pode ser usado ao trocar o ID BDK ou o KSI, mas não precisa trocar os outros dados contidos nos blocos IK e KS. Normalmente, o BI é usado ao transmitir para um KIF, enquanto IK ou KS são usados ao injetar no próprio terminal.
HM	Especifica o tipo de hash para operações 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 • 40 — SHAKE128 • 41 — SHAKE256	O serviço preenche automaticamente esse campo na exportação e o analisará na importação. Os tipos de hash não suportados pelo serviço, como o SHAKE128, podem ser importados, mas podem não ser usados para funções criptográficas.

Nome do cabeçalho	Finalidade	Validação típica	Observações
IK	Número de série da chave inicial para AES DUKPT	16 caracteres hexadecimais	Esse valor é usado para instanciar o uso da chave DUKPT inicial no dispositivo receptor e identifica a chave inicial derivada de um BDK. Esse campo normalmen te contém os dados de derivação, mas nenhum contador. Use KS para TDES DUKPT.
KS	Número de série da chave inicial para TDES DUKPT	20 caracteres hexadecimais	Esse valor é usado para instanciar o uso da chave DUKPT inicial no dispositivo receptor e identifica a chave inicial derivada de um BDK. Esse campo normalmen te contém os dados de derivação mais um valor de contador zerado. Use IK para AES DUKPT.

Nome do cabeçalho	Finalidade	Validação típica	Observações
KP	Valor de verificação da chave do KBPK	2 caracteres hexadecimais representam o método KCV (00 para o X9.24 método e 01 para o método CMAC). Seguido pelo valor KCV, que normalmente é de 6 caracteres hexadecimais. Por exemplo, 010FA329 representa o KCV de 0FA329 calculado usando o método 01 (CMAC).	Calculado de acordo com o X9.24-1-2017 Anexo A da ANSI; o KCV é usado para validação de chaves e não como um mecanismo de integridade.
PB	Bloco de preenchimento	caracteres ASCII imprimíveis aleatórios	O serviço preenche automaticamente esse campo na exportação para garantir que os cabeçalhos opcionais sejam múltiplos do comprimento do bloco de criptografia

Exportar chaves assimétricas (RSA)

Para exportar uma chave pública em formato de certificado, use o get-public-key-certificate comando. Esse comando retorna:

- O certificado
- O certificado raiz

Ambos os certificados estão na codificação base64.

> **Note**
>
> Essa operação não é idempotente — chamadas subsequentes podem gerar certificados diferentes, mesmo usando a mesma chave subjacente.

Example

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

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

Tópicos avançados

Esta seção aborda cenários e configurações avançadas de troca de chaves.

Tópicos

- Traga sua própria autoridade de certificação (BYOCA)
- Troca de chaves físicas

Traga sua própria autoridade de certificação (BYOCA)

Por padrão, quando um certificado de chave pública é necessário para chaves assimétricas (RSA, ECC) criadas no serviço, esses certificados são emitidos por uma Criptografia de AWS Pagamento e uma autoridade de certificação (CA) exclusiva da conta. O objetivo é simplificar o uso X.509 sem o ônus de identificar ou configurar uma CA ou gerenciar solicitações de assinatura de certificados (CSR).

AWS A criptografia de pagamento também permite usar sua própria CA quando necessário por motivos de política ou conformidade.

Tópicos

- Visão geral do
- Fluxo de trabalho de exportação BYOCA
- Fluxo de trabalho de importação BYOCA
- Notas adicionais

Visão geral do

Com o BYOCA, você pode usar sua própria CA para transferências de TR-34 import/export chaves, RSA Unwrap e ECDH. Use o BYOCA quando precisar de uma cadeia de certificados consistente em toda a sua organização ou quando os parceiros precisarem de certificados CA específicos. Os exemplos a seguir mostram o fluxo de trabalho BYOCA para TR-34 exportação e importação de TR-34 chaves.

Há três diferenças principais em relação ao TR-34 fluxo padrão:

1. Você cria a chave RSA com CreateKey. Antes, `GetParametersForExport` ou `GetParametersForImport` criei para você.
2. A GetCertificateSigningRequest API cria uma CSR. Sua CA externa pode então assiná-la.
3. As ImportKey APIs ExportKey e aceitam um certificado no momento da chamada. O token agora é opcional.

⚠ Considerações importantes

- Esses exemplos usam RSA-2048 chaves e envolvem uma TDES-2KEY chave. Ao exportar AES-128, certifique-se de que todas as chaves sejam RSA-3072 ou RSA-4096.
- O erro mais comum é que a chave representada por `SigningKeyIdentifier` e `SigningKeyCertificate` não coincide.

Fluxo de trabalho de exportação BYOCA

As etapas a seguir demonstram o fluxo de trabalho completo do BYOCA para TR-34 exportação.

Etapas

- Etapa 1: criar uma chave RSA
- Etapa 2: gerar solicitação de assinatura de certificado
- Etapa 3: Analisar a CSR (opcional)
- Etapa 4: assinar o CSR com uma autoridade de certificação
- Etapa 5: Importar o certificado CA
- Etapa 6: Obter o certificado de criptografia KRD
- Etapa 7: Exportar chave com BYOCA

Etapa 1: criar uma chave RSA

Primeiro, crie um par de chaves RSA que, em última instância, será o certificado de assinatura KDH. Você pode adicionar tags para identificar a finalidade da chave.

Example Criar chave RSA para assinatura

```
$ 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"
    }
}
```

Anote o `KeyArn` que você precisará na próxima etapa.

Etapa 2: gerar solicitação de assinatura de certificado

Gere uma Solicitação de Assinatura de Certificado (CSR) para ser assinada por sua CA externa usando a GetCertificateSigningRequest API. A saída é um arquivo PEM codificado em base64. Se você decodificar o conteúdo em base64 e salvá-lo, terá uma CSR válida no formato PEM.

Example Gerar CSR

```
$ aws payment-cryptography 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..."
}
```

O `CertificateSigningRequest` campo contém todo o arquivo PEM, codificado em base64. O arquivo PEM inclui os `-----END CERTIFICATE REQUEST-----` marcadores `-----BEGIN CERTIFICATE REQUEST-----` e. Envie esse valor para sua CA para assinatura.

Etapa 3: Analisar a CSR (opcional)

Opcionalmente, você pode usar o OpenSSL para revisar o conteúdo da CSR e garantir que ele seja válido e conforme o esperado.

Example Analise a CSR com o OpenSSL

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

Etapa 4: assinar o CSR com uma autoridade de certificação

Depois de gerar a CSR, você precisa que ela seja assinada por uma Autoridade Certificadora (CA). Em ambientes de produção, você normalmente CA privada da AWS usaria a infraestrutura de CA estabelecida da sua organização. Para fins de teste, você pode usar o OpenSSL para criar um certificado autoassinado.

Utilizar CA privada da AWS

Para assinar a CSR usando CA privada da AWS, primeiro decodifique a CSR codificada em base64 e salve-a em um arquivo e, em seguida, use a API. IssueCertificate

Example Assine CSR com CA privada da AWS

```
$ 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"
}
```

Em seguida, recupere o certificado assinado:

Example Recuperar certificado assinado

```
$ 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-----"
```

```
}
```

Salve o conteúdo do certificado para uso na etapa de exportação. Você precisará codificá-lo em base64 ao fornecê-lo à API. `ExportKey`

Usando o OpenSSL para testes

Para fins de teste, você pode usar o OpenSSL para criar uma CA autoassinada e assinar a CSR. Primeiro, crie uma chave privada da CA e um certificado autoassinado:

Example Crie uma CA de teste com o 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"
```

Em seguida, decodifique a CSR da etapa anterior e assine-a com sua CA de teste:

Example Assine CSR com 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
```

O certificado assinado está agora disponível`signed-cert.pem`. Você precisará codificar esse certificado em base64 ao fornecê-lo à API: `ExportKey`

Example Base64 Codifique o certificado assinado

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

Etapa 5: Importar o certificado CA

Qualquer CA usada precisa ser confiável primeiro para evitar que certificados arbitrários sejam usados. Importe o certificado raiz da sua CA externa usando a ImportKey API. Se estiver usando uma CA intermediária, chame `import-key` novamente, mas especifique `TrustedPublicKey` em vez `RootCertificatePublicKey` de especificar o ARN da CA raiz.

Example Importar certificado CA raiz

```
$ 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"
    }
}
```

Anote as CAs `KeyArn` para uso na etapa de exportação.

Etapa 6: Obter o certificado de criptografia KRD

Neste exemplo, estamos importando novamente para a criptografia de AWS pagamento, então chamamos o serviço para receber um certificado de chave pública KRD usando a API. GetParametersForImport Em um cenário real, isso seria fornecido por outro sistema, como um HSM, um caixa eletrônico, um terminal de pagamento ou um sistema de gerenciamento de terminal de pagamento.

Example Obter parâmetros a serem importados

```
$ aws payment-cryptography 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"
}
```

Etapa 7: Exportar chave com BYOCA

Por fim, exporte a chave usando TR-34 seu próprio CA-signed certificado usando a ExportKey API. Forneça o certificado de assinatura que foi assinado pela sua CA externa.

Example TR-34 Exportar com BYOCA

```
$ aws payment-cryptography 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"
    }
}
```

O bloco de chaves exportado agora pode ser importado pelo sistema receptor usando o processo de TR-34 importação padrão.

Fluxo de trabalho de importação BYOCA

As etapas a seguir orientam você pelo fluxo de trabalho BYOCA para importação de TR-34 chaves. Use esse fluxo de trabalho para receber uma chave na criptografia AWS de pagamento com seu próprio certificado de criptografia CA-signed KRD (Key Receiving Device). Com a importação BYOCA, você controla a cadeia de confiança usando uma CA que você já gerencia.

Etapas

- Etapa 1: criar uma chave RSA
- Etapa 2: gerar solicitação de assinatura de certificado
- Etapa 3: Analisar a CSR (opcional)
- Etapa 4: assinar o CSR com uma autoridade de certificação
- Etapa 5: Importar o certificado CA
- Etapa 6: Obter o certificado de assinatura KDH do sistema de envio
- Etapa 7: Importar chave com BYOCA

Etapa 1: criar uma chave RSA

Crie um par de chaves RSA para usar como chave de criptografia KRD. Essa chave desembrulha e descriptografa a chave recebida durante a importação. TR-34 A chave não é exportável porque permanece no serviço como chave receptora.

Example Criar chave de criptografia KRD

```
$ aws payment-cryptography create-key \
    --key-attributes
 KeyAlgorithm=RSA_2048,KeyUsage=TR31_K2_TR34_ASYMMETRIC_KEY,KeyClass=ASYMMETRIC_KEY_PAIR,KeyMod
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
bm7t4qv8hzk24jce",
        "KeyAttributes": {
            "KeyUsage": "TR31_K2_TR34_ASYMMETRIC_KEY",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyAlgorithm": "RSA_2048",
            "KeyModesOfUse": {
                "Unwrap": true,
                "Decrypt": true
            }
        },
        "KeyCheckValue": "7FA29C1E",
        "KeyCheckValueAlgorithm": "SHA_1",
        "Enabled": true,
        "Exportable": false,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY"
    }
}
```

Observe o valor `KeyArn`. Você precisa dele na próxima etapa.

Etapa 2: gerar solicitação de assinatura de certificado

Gere uma Solicitação de Assinatura de Certificado (CSR) para a chave de criptografia KRD usando a GetCertificateSigningRequest API. A saída é um arquivo PEM codificado em base64. Se você decodificar o conteúdo em base64 e salvá-lo, terá uma CSR válida no formato PEM.

Example Gerar CSR para a chave KRD

```
$ aws payment-cryptography get-certificate-signing-request \
    --key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
bm7t4qv8hzk24jce \
    --signing-algorithm SHA512 \
    --certificate-subject '{
```

```
        "CommonName": "MyImportKRDCertAWSUSEAST",
        "Organization": "Amazon",
        "OrganizationUnit": "PaymentCryptography",
        "Country": "US",
        "StateOrProvince": "Virginia",
        "City": "Arlington"
    }'
```

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

O `CertificateSigningRequest` campo contém todo o arquivo PEM, codificado em base64. O arquivo PEM inclui os `-----END CERTIFICATE REQUEST-----` marcadores `-----BEGIN CERTIFICATE REQUEST-----` e. Envie esse valor para sua CA para assinatura.

Etapa 3: Analisar a CSR (opcional)

Opcionalmente, você pode usar o OpenSSL para revisar o conteúdo da CSR e garantir que ele seja válido e conforme o esperado.

Example Analise a CSR com o OpenSSL

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

Etapa 4: assinar o CSR com uma autoridade de certificação

Depois de gerar a CSR, faça com que ela seja assinada por uma Autoridade Certificadora (CA). Em ambientes de produção, você normalmente CA privada da AWS usaria a infraestrutura de CA estabelecida da sua organização. Para fins de teste, você pode usar o OpenSSL para criar um certificado autoassinado.

Utilizar CA privada da AWS

Para assinar a CSR usando CA privada da AWS, primeiro decodifique a CSR codificada em base64 e salve-a em um arquivo e, em seguida, use a API. IssueCertificate

Example Assine CSR com CA privada da AWS

```
$ echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d > krd-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://krd-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/fedcba0987654321"
}
```

Em seguida, recupere o certificado assinado:

Example Recuperar certificado assinado

```
$ 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/fedcba0987654321
```

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

Salve o conteúdo do certificado para uso na etapa de importação. Base64-encode o certificado antes de fornecê-lo à `ImportKey` API.

Usando o OpenSSL para testes

Para fins de teste, você pode usar o OpenSSL para criar uma CA autoassinada e assinar a CSR. Primeiro, crie uma chave privada da CA e um certificado autoassinado:

Example Crie uma CA de teste com o 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"
```

Em seguida, decodifique a CSR da etapa anterior e assine-a com sua CA de teste:

Example Assine CSR com OpenSSL

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

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

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

O certificado assinado está agora disponívelkrd-signed-cert.pem. Base64-encode este certificado antes de fornecê-lo à ImportKey API:

Example Base64 Codifique o certificado assinado

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

Etapa 5: Importar o certificado CA

Antes de usar uma CA, você deve confiar nela primeiro. Importe o certificado raiz da sua CA externa usando a ImportKey API. Se estiver usando uma CA intermediária, chame import-key novamente, mas especifique TrustedPublicKey em vez RootCertificatePublicKey de especificar o ARN da CA raiz.

Example Importar certificado CA raiz

```
$ 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/
nqr3c5v2wp6yl8az",
        "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"
    }
}
```

Anote as CAs `KeyArn` para uso na etapa de importação.

Etapa 6: Obter o certificado de assinatura KDH do sistema de envio

Em uma troca de TR-34 chaves, o lado remetente (Key Distribution Host, ou KDH) fornece seu certificado de chave pública para assinatura. Esse certificado é usado para verificar a assinatura no bloco de chaves encapsulado durante a importação, garantindo que o material da chave tenha vindo de um remetente confiável. Na produção, esse certificado normalmente vem do sistema de envio, como um HSM, um terminal de pagamento ou um sistema de gerenciamento de chaves, por meio de seu próprio processo de distribuição de certificados.

Neste exemplo, você usa a criptografia AWS de pagamento como remetente e destinatário para manter as etapas curtas. Ligue GetParametersForExport para obter o certificado de assinatura KDH que normalmente seria fornecido pelo sistema de envio.

Example Obter parâmetros para exportação

```
$ aws payment-cryptography get-parameters-for-export \
```

```
    --key-material-type "TR34_KEY_BLOCK" \
    --signing-key-algorithm RSA_2048
```

```
{
    "ExportToken": "export-token-k8sfhe2w9rjy4bx6",
    "SigningKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "SigningKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "SigningKeyAlgorithm": "RSA_2048",
    "ParametersValidUntilTimestamp": "2025-11-01T18:45:31.271000-07:00"
}
```

Anote o `ExportToken` e `SigningKeyCertificate` para uso na etapa de importação.

Etapa 7: Importar chave com BYOCA

Por fim, importe a chave usando TR-34 seu próprio CA-signed certificado usando a `ImportKey` API. Forneça o certificado de criptografia KRD que foi assinado pela sua CA externa.

O `RandomNonce` parâmetro é necessário ao usar 2 passagens TR-34. O serviço não gera um nonce para você — é um valor hexadecimal aleatório que você gera e deve ser o mesmo no lado do envio e do recebimento. Ao usar 1-pass TR-34 (que depende de um carimbo de data/hora em vez de um nonce), o carimbo de data/hora não deve estar no futuro. O serviço não valida nenhuma atualização específica (por exemplo, dentro de um determinado número de horas), mas você pode impor verificações de atualização de sua parte, se necessário.

Example TR-34 Importe com BYOCA

```
$ aws payment-cryptography import-key --key-material='{
    "Tr34KeyBlock": {
        "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/nqr3c5v2wp6yl8az",
        "KeyBlockFormat": "X9_TR34_2012",
        "RandomNonce": "4997FBB4587D571F",
        "SigningKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
        "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
        "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/bm7t4qv8hzk24jce",
        "WrappedKeyBlock": "D0112B0AX00E0000B82679114F470F540165EDFBF407..."
    }
}'
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
gwz7u3rfxd4o52sb",
        "KeyAttributes": {
            "KeyUsage": "TR31_K0_KEY_ENCRYPTION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_2KEY",
            "KeyModesOfUse": {
                "Encrypt": true,
                "Decrypt": true
            }
        },
        "KeyCheckValue": "3DCA31",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL"
    }
}
```

A chave agora está importada para a criptografia AWS de pagamento e está pronta para uso. O `KeyOrigin` campo mostra a `EXTERNAL` indicação de que a chave veio de fora do serviço.

Notas adicionais

- Esses exemplos são mostrados usando o AWS CLI. A mesma funcionalidade está disponível em todos os SDKs da AWS, incluindo Java, Python, Go e Rust.
- Se você estiver testando com uma CA autoassinada, poderá usar o OpenSSL para criar uma CA de teste e assinar a CSR. Na produção, use a infraestrutura de CA estabelecida da sua organização.

Troca de chaves físicas

Você pode usar o Physical Key Exchange para converter com segurança componentes de chaves criptográficas em papel em formato eletrônico quando seus parceiros ou fornecedores não oferecem suporte à troca eletrônica de chaves. Guardiões de AWS chaves treinados realizam cerimônias importantes em instalações AWS-operated seguras com certificação PCI PIN e P2PE, convertendo os principais componentes em papel em formato eletrônico usando um HSM offline. O serviço usa a

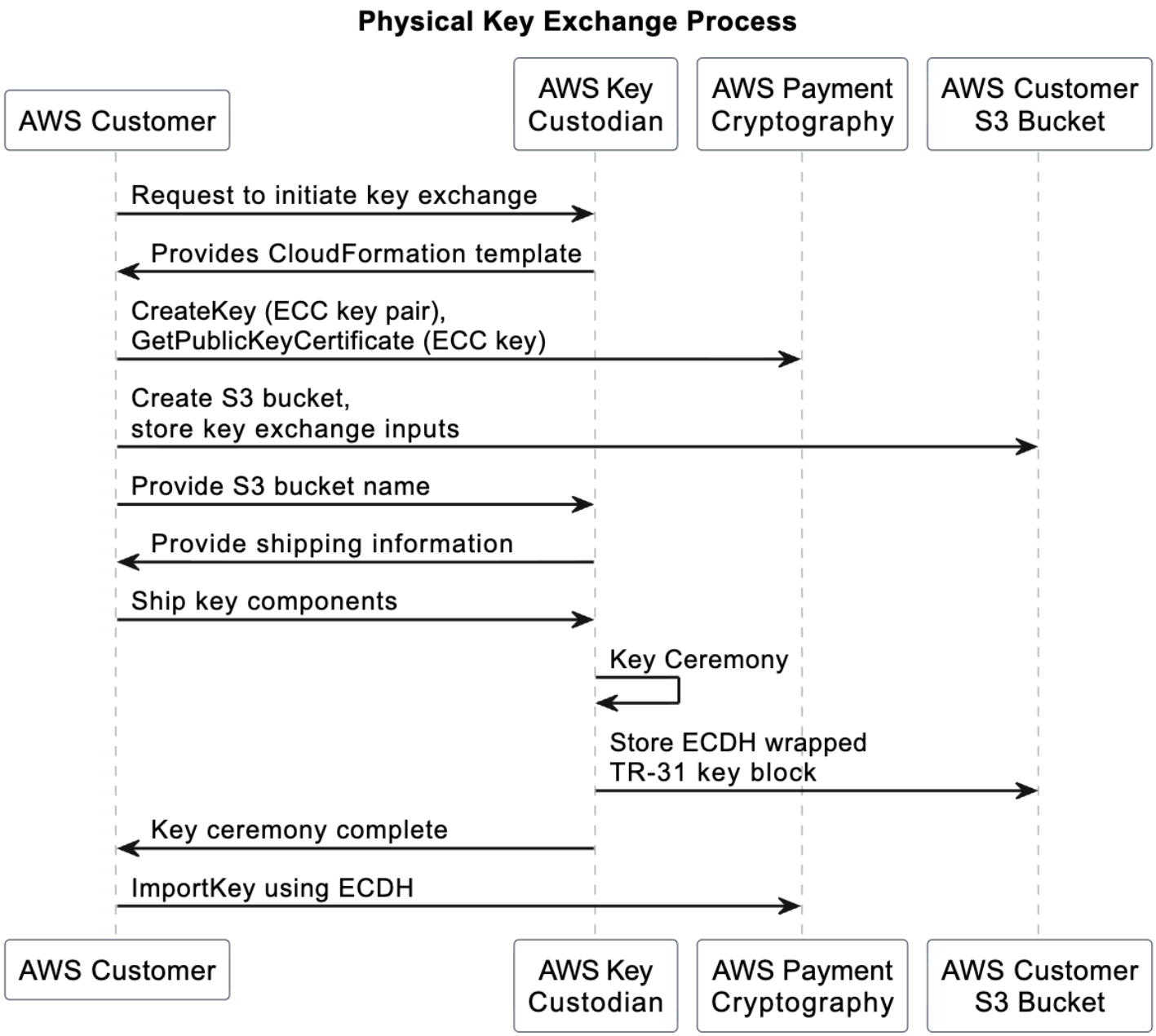
troca de ECDH-based chaves para entregar um bloco de ECDH-wrapped TR-31 chaves, que você importa diretamente para sua conta AWS de criptografia de pagamento.

> **Note**
>
> Recomendamos o uso baseado em padrões sempre Importando e exportando chaves que possível. Use o Physical Key Exchange somente quando seus parceiros ou fornecedores não oferecerem suporte a métodos de troca eletrônica de chaves, como ANSI X9.24 TR-34, RSA wrap/unwrap ou ECDH. ???

Como funciona o Physical Key Exchange

Para iniciar a troca de chaves em papel, um CloudFormation modelo orienta você na configuração dos pré-requisitos, incluindo a criação de um par de chaves ECC e um bucket S3 em sua conta. Em seguida, você ou seu parceiro enviam os principais componentes em papel para a instalação AWS segura, onde guardiões de AWS chaves treinados realizam a cerimônia principal usando um HSM offline. A saída é um bloco de ECDH-wrapped TR-31 chaves enviado para seu bucket do S3, que você importa para sua conta usando o Importar chaves usando técnicas assimétricas (ECDH) método. O Physical Key Exchange suporta a importação de chaves KEK (uso da chave K1) ou BDK (uso da chave B0) nos algoritmos de chave TDES e AES.

O diagrama a seguir mostra o processo de troca de chaves físicas de ponta a ponta.



1. Iniciação — Você envia um ticket de suporte ou trabalha com seu gerente de conta para enviar uma solicitação.
2. Configuração do cliente — A criptografia de AWS pagamento fornece um CloudFormation modelo para você concluir as seguintes etapas de pré-requisito:
   - Crie um par de chaves ECC P521 em sua conta de criptografia de AWS pagamento e recupere o certificado de chave pública.

- Crie um bucket do Amazon S3 com uma política que conceda acesso principal ao serviço AWS de criptografia de pagamento. read/write
- Armazene o certificado público ECC e a CA raiz de assinatura no bucket do Amazon S3.
- Forneça os principais atributos: uso da chave, principais modos de uso e número de componentes principais em papel a serem enviados.

3. Compartilhe o nome do bucket do S3 — O cliente compartilha o nome do bucket do S3 criado pela CloudFormation pilha, onde o certificado de chave pública, a cadeia de certificados e os atributos da chave são armazenados para que a criptografia de AWS pagamento inicie a troca de chaves.
4. Coordenação de envio — A criptografia de AWS pagamento fornece detalhes de envio para a instalação US-based segura. Você ou seu parceiro enviam os principais componentes em papel para os AWS principais guardiões.
5. Recibo do componente — os AWS principais guardiões recebem cada componente em papel e enviam uma confirmação separada para cada componente.
6. Cerimônia da AWS chave — os principais guardiões realizam a cerimônia da chave usando um HSM offline. O bloco de TR-31 chaves resultante, encapsulado usando uma ECDH-derived AES-256 chave, o certificado público ECC do HSM off-line e seu certificado de assinatura são enviados para seu bucket do Amazon S3.
7. Conclusão — A criptografia de AWS pagamento envia uma confirmação de que a cerimônia principal foi concluída. Você pode então importar o bloco de TR-31 chaves encapsulado em ECDH para sua conta AWS de criptografia de pagamento usando o método. Importar chaves usando técnicas assimétricas (ECDH)
8. Cobrança — Você é cobrado por chave trocada após a conclusão bem-sucedida da cerimônia principal.

Segurança e conformidade

O Physical Key Exchange opera em instalações AWS seguras projetadas para atender aos requisitos de segurança física e lógica do PCI PIN e do PCI P2PE. Os seguintes controles estão em vigor:

Controle duplo e separação de funções

AWS os principais guardiões são designados por equipes diferentes com estruturas de relatórios separadas. Os processos estão em vigor para garantir que as principais etapas das cerimônias sejam realizadas sob controle duplo.

HSM off-line

As principais cerimônias são realizadas usando módulos de segurança de HSM-listed hardware PCI PTS certificados que operam off-line sem conectividade de rede. Sua chave nunca existe em texto não criptografado fora dos limites do HSM.

Entrega de chave criptográfica

O material da chave é transferido do HSM off-line para sua conta de criptografia AWS de pagamento usando a troca de ECDH-based chaves, garantindo proteção criptográfica de ponta a ponta.

Auditoria e conformidade

AWS tem processos implementados para atender aos requisitos de conformidade aplicáveis que são avaliados periodicamente para os atestados PCI PIN e P2PE. Consulte o pacote de conformidade no AWS Artifact para ver os relatórios que você tem como referência em suas próprias avaliações de PCI.

Usar aliases

Um alias é um nome amigável para uma chave de criptografia AWS de pagamento. Por exemplo, um alias permite fazer referência a uma chave como `alias/test-key` em vez de `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h`.

Você pode usar um alias para identificar uma chave na maioria das operações de gerenciamento de chaves (plano de controle) e em operações criptográficas (plano de dados).

Você também pode permitir e negar acesso à chave AWS de criptografia de pagamento com base em seus aliases sem editar políticas ou gerenciar subsídios. Esse atributo faz parte do suporte do serviço para controle de acesso por atributo (ABAC).

Aliases também facilitam a reutilização do mesmo código em Regiões da AWS diferentes. Crie aliases com o mesmo nome em várias regiões e associe cada alias a uma chave de criptografia AWS de pagamento em sua região. Quando o código é executado em cada região, o alias se refere à chave de criptografia AWS de pagamento associada nessa região.

Você pode criar um alias para uma chave AWS de criptografia de pagamento usando a `CreateAlias` API.

A API AWS de criptografia de pagamento fornece controle total dos aliases em cada conta e região. A API inclui operações para criar um alias (CreateAlias), visualizar nomes de alias e o keyARN vinculado (list-aliases), alterar a chave de criptografia de AWS pagamento associada a um alias (update-alias) e excluir um alias (delete-alias).

Tópicos

- Sobre aliases
- Usar aliases em suas aplicações
- APIs relacionadas

Sobre aliases

Saiba como os aliases funcionam na criptografia AWS de pagamentos.

Um alias é um recurso independente AWS

Um alias não é uma propriedade de uma chave de criptografia AWS de pagamento. As ações executadas no alias não afetam a chave associada. Você pode criar um alias para uma chave de criptografia de AWS pagamento e depois atualizar o alias para que ele seja associado a uma chave de criptografia de AWS pagamento diferente. Você pode até mesmo excluir o alias sem qualquer efeito na chave de criptografia AWS de pagamento associada. Se você excluir uma chave do AWS Payment Cryptography, todos os aliases associados a essa chave deixarão de ser atribuídos.

Se você especificar um alias como recurso em uma política do IAM, a política se referirá ao alias, não à chave de criptografia de AWS pagamento associada.

Cada alias tem um nome fácil de usar

Ao criar um alias, você especifica o nome do alias prefixado por `alias/`. Por exemplo, `alias/test_1234`

Cada alias é associado a uma chave AWS de criptografia de pagamento por vez

O alias e sua chave AWS de criptografia de pagamento devem estar na mesma conta e região.

Uma chave AWS de criptografia de pagamento pode ser associada a mais de um alias simultaneamente, mas cada alias só pode ser mapeado para uma única chave

Por exemplo, esta saída de `list-aliases` mostra que o alias `alias/sampleAlias1` está associado exatamente a uma chave de AWS Payment Cryptography de destino, que é representada pela propriedade `KeyArn`.

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

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

Vários aliases podem ser associados à mesma chave de criptografia AWS de pagamento

Por exemplo, você pode associar os aliases `alias/sampleAlias1;` e `alias/sampleAlias2` à mesma chave.

```
$ 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"
            }
        ]
```

```
}
```

Um alias deve ser exclusivo para uma determinada conta e região

Por exemplo, é possível ter apenas um alias `alias/sampleAlias1` em cada conta e região. Os aliases diferenciam maiúsculas de minúsculas, mas não recomendamos usar aliases que diferem apenas no tamanho das letras, pois podem estar propensos a erros. Não é possível alterar um nome de alias. No entanto, você pode excluir o alias e criar um novo com o nome desejado.

É possível criar um alias com o mesmo nome em regiões diferentes

Por exemplo, você pode ter um alias `alias/sampleAlias2` no Leste dos EUA (Norte da Virgínia) e um alias `alias/sampleAlias2` no Oeste dos EUA (Oregon). Cada alias seria associado a uma chave de criptografia AWS de pagamento em sua região. Se o seu código se referir a um nome de alias como `alias/finance-key`, você poderá executá-lo em várias regiões. Em cada região, ele usa uma alias/sampleAlias2 diferente. Para obter detalhes, consulte Usar aliases em suas aplicações.

Você pode alterar a chave AWS de criptografia de pagamento associada a um alias

Você pode usar a `UpdateAlias` operação para associar um alias a uma chave de criptografia AWS de pagamento diferente. Por exemplo, se o `alias/sampleAlias2` alias estiver associado à chave de criptografia de `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` AWS pagamento, você poderá atualizá-lo para que fique associado à `arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi` chave.

> **Warning**
>
> AWS A criptografia de pagamento não valida que as chaves antigas e novas tenham todos os mesmos atributos, como o uso da chave. A atualização com um tipo de chave diferente pode resultar em problemas em seu aplicativo.

> **Warning**
>
> As atualizações dos aliases estão sujeitas a uma eventual consistência. Depois de atualizar um alias, há um breve período durante o qual o alias pode ser resolvido para a chave antiga ou para a nova chave. Não atualize aliases se esse comportamento interromper seus processos.

Algumas chaves não têm aliases

Um alias é um atributo opcional e nem todas as chaves terão aliases, a menos que você opte por operar seu ambiente dessa maneira. As chaves podem ser associadas a aliases usando o comando `create-alias`. Além disso, você pode usar a operação update-alias para alterar a chave de AWS Payment Cryptography associada a um alias e a operação delete-alias para excluir um alias. Como resultado, algumas chaves AWS de criptografia de pagamento podem ter vários aliases e outras podem não ter nenhum.

Mapear uma chave para um alias

É possível mapear uma chave (representada por um ARN) para um ou mais aliases usando o comando `create-alias`. Esse comando não é idempotente. Para atualizar um alias, use o comando 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"
    }
}
```

Usar aliases em suas aplicações

Você pode usar um alias para representar uma chave de criptografia AWS de pagamento no código do seu aplicativo. O `key-identifier` parâmetro nas operações de dados AWS de criptografia de pagamento, bem como em outras operações, como List Keys, aceita um nome de alias ou um ARN de alias.

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

Ao usar um alias ARN, lembre-se de que o mapeamento do alias para uma chave de criptografia de AWS pagamento é definido na conta que possui a chave de criptografia de AWS pagamento e pode ser diferente em cada região.

Um dos usos mais poderosos dos aliases é em aplicações executadas em várias Regiões da AWS.

Você pode criar uma versão diferente do seu aplicativo em cada região ou usar um dicionário, configuração ou declaração de switch para selecionar a chave de criptografia de AWS pagamento correta para cada região. Mas pode ser mais fácil criar um alias com o mesmo nome em cada região. Lembre-se de que o nome do alias diferencia maiúsculas de minúsculas.

APIs relacionadas

Tags

As tags são pares de chaves e valores que funcionam como metadados para organizar suas chaves de criptografia AWS de pagamento. Elas podem ser usadas para identificar chaves de forma flexível ou agrupar uma ou mais chaves.

Obter chaves

Uma chave AWS de criptografia de pagamento representa uma única unidade de material criptográfico e só pode ser usada para operações criptográficas desse serviço. A GetKeys API usa um KeyIdentifier como entrada e retorna os metadados da chave, incluindo atributos, estado e carimbos de data/hora, mas não retorna o material real da chave criptográfica.

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 Associe o público a um par de chaves

Get Public Key/Certificate retorna a chave pública indicada pelo`KeyArn`. Essa pode ser a parte da chave pública de um par de chaves gerado na criptografia AWS de pagamento ou uma chave pública importada anteriormente. O caso de uso mais comum é fornecer a chave pública a um serviço externo que fará a criptografia dos dados. Esses dados podem então ser passados para um aplicativo que utiliza a criptografia de AWS pagamento e os dados podem ser descriptografados usando a chave privada protegida pela criptografia de pagamento. AWS

O serviço retorna as chaves públicas como um certificado público. O resultado da API contém a CA e o certificado de chave pública. Ambos os elementos de dados são codificados em base64.

> **Note**
>
> O certificado público retornado deve durar pouco e não ser idempotente. É possível receber um certificado diferente em cada chamada de API, mesmo que a chave pública permaneça inalterada.

Example

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

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

Marcar chaves com tags

Na Criptografia de AWS pagamento, você pode adicionar tags a uma chave de criptografia de AWS pagamento ao criar uma chave e marcar ou desmarcar as chaves existentes, a menos que elas estejam pendentes de exclusão. Etiquetas são opcionais, mas podem ser bastante úteis.

Para obter informações gerais sobre tags, incluindo as melhores práticas, estratégias de marcação e o formato e a sintaxe das tags, consulte AWS Recursos de marcação no. Referência geral da Amazon Web Services

Tópicos

- Sobre tags em AWS Payment Cryptography
- Visualizar tags de chave no console
- Gerenciar tags de chave com operações de API
- Controlar o acesso às tags
- Usar tags para controlar o acesso a chaves

Sobre tags em AWS Payment Cryptography

Uma tag é um rótulo de metadados opcional que você pode atribuir (ou AWS atribuir) a um AWS recurso. Cada tag consiste em uma chave de tag e um valor de tag, sendo ambos strings que diferenciam maiúsculas de minúsculas. O valor da tag pode ser uma string vazia (nula). Cada tag em um recurso deve ter uma chave de tag diferente, mas você pode adicionar a mesma tag a vários AWS recursos. Cada recurso pode ter até 50 tags criadas pelo usuário.

Não inclua informações confidenciais ou sigilosas na chave ou no valor da tag. As tags podem ser acessadas por muitos Serviços da AWS, incluindo o faturamento.

Na Criptografia de AWS pagamento, você pode adicionar tags a uma chave ao criar a chave e marcar ou desmarcar as chaves existentes, a menos que elas estejam pendentes de exclusão. Não é possível marcar aliases com tags. Etiquetas são opcionais, mas podem ser bastante úteis.

Por exemplo, você pode adicionar uma `"Project"="Alpha"` tag a todas as chaves de criptografia AWS de pagamento e buckets do Amazon S3 que você usa para o projeto Alpha. Outro exemplo é adicionar uma tag `"BIN"="20130622"` a todas as chaves associadas a um número de identificação bancária (BIN) específico.

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

Para obter informações gerais sobre tags, incluindo o formato e a sintaxe, consulte AWS Recursos de marcação no. Referência geral da Amazon Web Services

As tags ajudam a:

- Identifique e organize seus AWS recursos. Muitos AWS serviços oferecem suporte à marcação, então você pode atribuir a mesma tag a recursos de serviços diferentes para indicar que os recursos estão relacionados. Por exemplo, você pode atribuir a mesma tag a uma chave de criptografia AWS de pagamento e a um volume ou AWS Secrets Manager segredo do Amazon Elastic Block Store (Amazon EBS). Você também pode usar tags para identificar chaves para automação.
- Acompanhe seus AWS custos. Quando você adiciona tags aos seus AWS recursos, AWS gera um relatório de alocação de custos com uso e custos agregados por tags. Você pode usar esse recurso para rastrear os custos AWS de criptografia de pagamento de um projeto, aplicativo ou centro de custo.

  Para obter mais informações sobre como usar etiquetas para alocação de custos, consulte Usar etiquetas de alocação de custos no Manual do usuário do AWS Billing . Para obter informações sobre as regras para chaves e valores de tags, consulte Restrições de User-Defined tags no Guia AWS Billing do usuário.
- Controle o acesso aos seus AWS recursos. Permitir e negar acesso às chaves com base em suas tags faz parte do suporte à criptografia de AWS pagamento para controle de acesso baseado em atributos (ABAC). Para obter informações sobre como controlar o acesso à AWS Payment Cryptography com base em suas tags, consulte Autorização baseada em AWS Tags de criptografia de pagamento. Para obter mais informações gerais sobre o uso de tags para controlar o acesso aos AWS recursos, consulte Controlando o acesso aos AWS recursos usando tags de recursos no Guia do usuário do IAM.

AWS A criptografia de pagamento grava uma entrada em seu AWS CloudTrail registro quando você usa as ListTagsForResource operações TagResource, UntagResource, ou.

Visualizar tags de chave no console

Para visualizar tags no console, é necessário ter permissão para marcar com tags na chave de uma política do IAM que inclua a chave. Essas permissões são necessárias além das requeridas para visualizar chaves no console.

Gerenciar tags de chave com operações de API

É possível usar a API de AWS Payment Cryptography para adicionar, excluir e listar tags para as chaves que você gerencia. Estes exemplos usam a AWS Command Line Interface (AWS CLI), mas você pode usar qualquer linguagem de programação compatível. Você não pode marcar Chaves gerenciadas pela AWS.

Para adicionar, editar, visualizar e excluir tags de uma chave, é necessário ter as permissões apropriadas. Para obter detalhes, consulte Controlar o acesso às tags.

Tópicos

- CreateKey: Adicionar tags a uma nova chave
- TagResource: Adicionar ou alterar tags para uma chave
- ListResourceTags: Obtenha as tags de uma chave
- UntagResource: Excluir tags de uma chave

CreateKey: Adicionar tags a uma nova chave

Você pode adicionar tags ao criar uma chave. Para especificar as tags, use o `Tags` parâmetro da CreateKey operação.

Para adicionar tags ao criar uma chave, o chamador deve ter a permissão `payment-cryptography:TagResource` em uma política do IAM. Essa permissão deve abranger, no mínimo, todas as chaves na conta e na região. Para obter detalhes, consulte Controlar o acesso às tags.

O valor do parâmetro `Tags` de `CreateKey` é uma coleção de pares de chave de etiqueta e valor de etiqueta que faz distinção entre maiúsculas e minúsculas. Cada tag em uma chave deve ter um nome de tag diferente. O valor da tag pode ser uma string nula ou vazia.

Por exemplo, o AWS CLI comando a seguir cria uma chave de criptografia simétrica com uma `Project:Alpha` tag. Ao especificar mais de um par de chave-valor, use um espaço para separar cada par.

```
$ 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"}]'
```

Quando esse comando é bem-sucedido, ele retorna um objeto `Key` com informações sobre a nova chave. No entanto, o `Key` não inclui tags. Para obter as tags, use a ListResourceTags operação.

TagResource: Adicionar ou alterar tags para uma chave

A TagResource operação adiciona uma ou mais tags a uma chave. Não é possível usar essa operação para adicionar ou editar etiquetas em uma Conta da AWS diferente.

Para adicionar uma tag, especifique uma nova chave e um valor de tag. Para editar uma tag, especifique uma chave de tag existente e um novo valor de tag. Cada tag em uma chave deve ter uma chave de tag diferente. O valor da tag pode ser uma string nula ou vazia.

Por exemplo, o comando a seguir adiciona as tags **`UseCase`** e **`BIN`** a uma chave demonstrativa.

```
$ 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"}]'
```

Quando esse comando for executado com êxito, ele não retornará nenhuma saída. Para visualizar as tags em uma chave, use a ListResourceTags operação.

Você também pode usar TagResource para alterar o valor de uma tag existente. Para substituir um valor de tag, especifique a mesma chave de tag com um valor diferente. Tags não listadas em um comando de modificação não são alteradas nem removidas.

Por exemplo, esse comando altera o valor da tag `Project` de `Alpha` para `Noe`.

O comando retornará http/200 sem conteúdo. Para ver suas alterações, use `ListTagsForResource`

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

ListResourceTags: Obtenha as tags de uma chave

A ListResourceTags operação obtém as tags de uma chave. O parâmetro `ResourceArn` (keyARN ou keyAlias) é obrigatório. Essa operação não pode ser usada para visualizar as tags nas chaves em uma Conta da AWS diferente.

Por exemplo, o comando a seguir obtém as tags para uma chave demonstrativa.

```
$ 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: Excluir tags de uma chave

A UntagResource operação exclui tags de uma chave. Para identificar as etiquetas a serem excluídas, especifique as chaves de etiqueta. Essa operação não pode ser usada para excluir tags de chaves em uma Conta da AWS diferente.

Quando é bem-sucedida, a operação `UntagResource` não retorna nenhuma saída. Além disso, se a chave da tag especificada não for encontrada na chave, ela não lançará uma exceção nem retornará uma resposta. Para confirmar que a operação funcionou, use a ListResourceTags operação.

Por exemplo, esse comando exclui a tag **`Purpose`** e seu valor com base na chave especificada.

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

Controlar o acesso às tags

Para adicionar, visualizar e excluir tags usando a API, as entidades principais precisam de permissões de marcação nas políticas do IAM.

Você também pode limitar essas permissões usando chaves de condição AWS globais para tags. Na criptografia AWS de pagamento, essas condições podem controlar o acesso às operações de marcação, como e. TagResource UntagResource

Para mais informações e exemplos de políticas, consulte Controlar o acesso baseado em chaves de etiqueta, no Guia do Usuário do IAM.

As permissões para criar e gerenciar aliases funcionam como a seguir.

criptografia de pagamento: TagResource

Permite que as entidades principais adicionem ou editem etiquetas. Para adicionar tags ao criar uma chave, a entidade principal deve ter permissão em uma política do IAM que não esteja restrita a chaves específicas.

criptografia de pagamento: ListTagsForResource

Permite que as entidades principais visualizem tags em chaves.

criptografia de pagamento: UntagResource

Permite que as entidades principais excluam tags de chaves.

Permissões de etiquetas em políticas

Você pode fornecer permissões de marcação em uma política de chaves ou política do IAM. O seguinte exemplo de política de chaves concede permissão de marcação a usuários selecionados na chave. Ele concede a todos os usuários que podem assumir os exemplos de funções Administrador ou Desenvolvedor permissão para visualizar etiquetas.

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": "*"
    }
  ]
}
```

Para conceder permissão de marcação de entidades principais em várias chaves, é possível usar uma política do IAM. Para que essa política seja eficiente, a política de chaves de cada chave deve permitir que a conta utilize políticas do IAM para controlar o acesso à chave.

Por exemplo, a seguinte política do IAM permite que as entidades principais criem chaves. Ela também permite que eles criem e gerenciem tags em todas as chaves na conta especificada. Essa combinação permite que os diretores usem o parâmetro tags da CreateKey operação para adicionar tags a uma chave enquanto a criam.

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/*"
    }
  ]
}
```

Limitar permissões de etiquetas

É possível limitar permissões de marcação usando condições de política. As seguintes condições de política podem ser aplicadas às permissões `payment-cryptography:TagResource` e `payment-cryptography:UntagResource`. Por exemplo, você pode usar a condição `aws:RequestTag/`*`tag-key`* para permitir que uma entidade principal adicione apenas etiquetas

específicas, ou pode impedir que uma entidade principal adicione etiquetas com chaves de etiqueta específicas.

- foi: RequestTag
- aws:ResourceTag/tag-key (somente políticas do IAM)
- foi: TagKeys

Como prática recomendada ao usar tags para controlar o acesso a chaves, use as chaves de condição `aws:RequestTag/`*`tag-key`* ou `aws:TagKeys` para determinar quais tags (ou chaves de tag) são permitidas.

Por exemplo, a política do IAM a seguir é semelhante à anterior. No entanto, essa política permite que as entidades principais criem etiquetas (`TagResource`) e excluam etiquetas `UntagResource` somente para etiquetas com um chave de etiqueta `Project`.

Como `TagResource` as `UntagResource` solicitações e podem incluir várias tags, você deve especificar um operador `ForAllValues` ou `ForAnyValue` set com a TagKeys condição aws:. O operador `ForAnyValue` exige que pelo menos uma das chaves de etiqueta na solicitação corresponda a uma das chaves de etiqueta na política. O operador `ForAllValues` requer que todas as chaves de etiqueta na solicitação correspondam a uma das chaves de etiqueta na política. O `ForAllValues` operador também retorna `true` se não houver tags na solicitação, mas TagResource UntagResource falhará quando nenhuma tag for especificada. Para detalhes sobre os operadores de conjunto, consulteUsar várias chaves e valores, no Manual do usuário do 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"}
      }
    }
  ]
}
```

Usar tags para controlar o acesso a chaves

Você pode controlar o acesso à criptografia de AWS pagamento com base nas tags da chave. Por exemplo, você pode escrever uma política do IAM que permite que as entidades principais habilitem e desabilitem somente as chaves que possuem uma tag específica. Ou você pode usar uma política do IAM para impedir que as entidades principais usem chaves em operações de criptografia, a menos que a chave tenha uma tag específica.

Esse recurso faz parte do suporte à criptografia de AWS pagamento para controle de acesso baseado em atributos (ABAC). Para obter informações sobre o uso de tags para controlar o acesso aos AWS recursos, consulte Para AWS que serve o ABAC? e controlar o acesso aos AWS recursos usando tags de recursos no Guia do usuário do IAM.

AWS A criptografia de pagamento é compatível com a chave de contexto de condição global aws:ResourceTag/tag-key, que permite controlar o acesso às chaves com base nas tags da chave. Como várias chaves podem ter a mesma tag, esse atributo permite que você aplique a permissão a um conjunto selecionado de chaves. Também é possível alterar facilmente as chaves no conjunto alterando suas tags.

Na criptografia AWS de pagamento, a chave de `aws:ResourceTag/`*`tag-key`* condição é suportada somente nas políticas do IAM. Ele não é suportado em políticas de chaves, que se aplicam somente a uma chave, ou em operações que não usam uma chave específica, como as ListAliases operações ListKeys ou.

Controlar o acesso com etiquetas é uma maneira simples, escalável e flexível de gerenciar permissões. No entanto, se isso não for projetado e gerenciado corretamente, poderá permitir ou negar acesso às chaves inadvertidamente. Se estiver usando etiquetas para controlar o acesso, considere as seguintes práticas.

- Use tags para reforçar a prática recomendada do acesso acesso com privilégio mínimo. Conceda às entidades principais do IAM somente as permissões de que eles precisam nas chaves que elas devem usar ou gerenciar. Por exemplo, use tags para rotular as chaves usadas para um projeto. Em seguida, dê permissão à equipe do projeto para usar somente chaves com a tag do projeto.
- Tenha cuidado ao conceder às entidades principais as permissões `payment-cryptography:TagResource` e `payment-cryptography:UntagResource`, com as quais elas podem adicionar, editar e excluir etiquetas. Quando você usa tags para controlar o acesso a chaves, a alteração de uma tag pode dar permissão às entidades principais para usar chaves que, de outra forma, elas não teriam permissão de usar. Ele também pode negar acesso a chaves que outras entidades principais exigem para realizar seus trabalhos. Os administradores de chaves que não tiverem permissão para alterar políticas de chave ou criar concessões poderão controlar o acesso às chaves se tiverem permissão para gerenciar tags.

  Sempre que possível, use uma condição de política, como `aws:RequestTag/`*`tag-key`* ou `aws:TagKeys`, para limitar as permissões de marcação de uma entidade principal para tags ou padrões de tags específicos em chaves específicas.
- Analise os diretores Conta da AWS que atualmente têm permissões de marcação e desmarcação e ajuste-os, se necessário. As políticas do IAM podem conceder permissões de marcação ou desmarcação em todas as chaves. Por exemplo, a política gerenciada pelo administrador permite que as entidades principais marquem, desmarquem e listem tags em todas as chaves.
- Antes de definir uma política que depende de uma tag, revise as tags nas chaves do seu Conta da AWS. Certifique-se de que sua política se aplica somente às etiquetas que você pretende incluir. Use CloudTrail registros e CloudWatch alarmes para alertá-lo sobre alterações de tags que possam afetar o acesso às suas chaves.
- As condições de políticas baseadas em etiquetas usam correspondência de padrões. Elas não estão vinculadas a uma instância específica de uma etiqueta. Uma política que usa chaves de condição baseadas em etiquetas afeta todas as etiquetas novas e existentes que correspondem ao padrão. Se você excluir e recriar uma etiqueta que corresponde a uma condição de política, a condição se aplicará à nova etiqueta, assim como à antiga.

Por exemplo, considere a seguinte política do IAM. Isso permite que as entidades principais chamem as operações de Decrypt somente em chaves em sua conta que estejam na região Leste dos EUA (Norte da Virgínia) e tenham uma tag `"Project"="Alpha"`. Você pode anexar essa política a funções no exemplo do projeto 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"
        }
      }
    }
  ]
}
```

O exemplo de política do IAM a seguir permite que as entidades principais usem qualquer chave na conta para determinadas operações de criptografia. Porém, ela proíbe as entidades principais de usar estas operações criptográficas em chaves com uma tag `"Type"="Reserved"` ou sem a tag `"Type"`.

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"
        }
      }
    }
  ]
}
```

Compreendendo os principais atributos do AWS Chave de criptografia de pagamento

Um princípio do gerenciamento adequado de chaves é que as chaves têm um escopo adequado e podem ser usadas apenas para operações permitidas. Dessa forma, certas chaves só podem ser criadas com determinados modos de uso de chaves. Sempre que possível, isso se alinha aos modos de uso disponíveis, conforme definido por TR-31.

Embora a criptografia de AWS pagamento impeça que você crie chaves inválidas, combinações válidas são fornecidas aqui para sua conveniência.

Chaves simétricas

- TR31_B0_BASE_DERIVATION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_C0_CARD_VERIFICATION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY*, AES_128*, AES_192*, AES_256*
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E1_EMV_MKEY_CONFIDENTIALITY
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E2_EMV_MKEY_INTEGRITY
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*

  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E5_EMV_MKEY_CARD_PERSONALIZATION
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E6_EMV_MKEY_OTHER
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}, { NoRestrictions = true}
- TR31_K0_KEY_ENCRYPTION_KEY
  - É recomendável usar TR31_K1_KEY_BLOCK_PROTECTION_KEY. Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_K1_KEY_BLOCK_PROTECTION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {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
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY
  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M3_ISO_9797_3_MAC_KEY
  - Algoritmos de chave permitidos: TDES_2KEY, TDES_3KEY
  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M6_ISO_9797_5_CMAC_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256

  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M7_HMAC_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_P0_PIN_ENCRYPTION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_V1_IBM3624_PIN_VERIFICATION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_V2_VISA_PIN_VERIFICATION_KEY
  - Algoritmos de chave permitidos: TDES_2KEY ,TDES_3KEY ,AES_128 ,AES_192 ,AES_256
  - Combinação permitida dos principais modos de uso: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}

Chaves assimétricas

- TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION
  - Algoritmos de chave permitidos: RSA_2048 ,RSA_3072 ,RSA_4096
  - Combinação permitida dos principais modos de uso: { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } ,{ Encrypt = true, Wrap = true } ,{ Decrypt = true, Unwrap = true }
  - OBSERVAÇÃO: {Encrypt = true, Wrap = true} é a única opção válida ao importar uma chave pública destinada a criptografar dados ou agrupar uma chave
- TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE
  - Algoritmos de chave permitidos: RSA_2048 ,RSA_3072 ,RSA_4096
  - Combinação permitida dos principais modos de uso: {Sign = true}, {Verify = true}

  - OBSERVAÇÃO: {Verify = true} é a única opção válida ao importar uma chave destinada à assinatura, como certificado raiz, certificado intermediário ou certificados de assinatura para TR-34.
- TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT
  - Usado para algoritmos de acordo chave, como ECDH
  - Algoritmos de chave permitidos: ECC_NIST_P256, ECC_NIST_P384, ECC_NIST_P521
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}.
  - NOTA: DeriveKeyUsage é usado para especificar que tipo de chave será derivada dessa chave base. Isso é corrigido na chave creation/import.
- TR31_K2_TR34_CHAVE_ASSIMÉTRICA
  - Chave assimétrica usada para mecanismos de troca de chaves X9.24 compatíveis, como TR-34
  - Algoritmos de chave permitidos: RSA_2048, RSA_3072, RSA_4096
  - Combinação permitida dos principais modos de uso: { DeriveKey = true}.
  - Combinação permitida dos principais modos de uso: { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } ,{ Encrypt = true, Wrap = true } ,{ Decrypt = true, Unwrap = true }
  - OBSERVAÇÃO: {Encrypt = true, Wrap = true} é a única opção válida ao importar uma chave pública destinada a criptografar dados ou agrupar uma chave

* Atualmente, essa combinação algorithm/key de tipos não é suportada por nenhuma operação criptográfica

Operações de dados

Depois de estabelecer uma chave AWS de criptografia de pagamento, ela pode ser usada para realizar operações criptográficas. Operações diferentes realizam diferentes tipos de atividade, desde criptografia, hashing até algoritmos específicos de domínio, como geração de CVV2.

Os dados criptografados não podem ser descriptografados sem a chave de decodificação correspondente (a chave simétrica ou a chave privada, dependendo do tipo de criptografia). Da mesma forma, os algoritmos de hash e específicos de domínio não podem ser verificados sem a chave simétrica ou a chave pública.

Para obter informações sobre tipos de chaves válidas para operações específicas, consulte Chaves válidas para operações criptográficas

> **Note**
>
> Recomendamos o uso de dados de teste em um ambiente que não seja de produção. O uso de chaves e dados de produção (PAN, BDK ID etc.) em um ambiente que não seja de produção pode afetar seu escopo de conformidade, como PCI DSS e PCI P2PE.

Tópicos

- Criptografe, descriptografe e recriptografe dados
- Gerar e verificar dados do cartão
- Gerar, traduzir e verificar dados de PIN
- Gere e verifique o ARQC
- Gerar e verificar MAC
- Chaves válidas para operações criptográficas

Criptografe, descriptografe e recriptografe dados

Métodos de criptografia e descriptografia podem ser usados para criptografar ou descriptografar dados usando uma variedade de técnicas simétricas e assimétricas, incluindo TDES, AES e RSA. Esses métodos também oferecem suporte a chaves derivadas usando as técnicas DUKPT e EMV. Para casos de uso em que você deseja proteger dados com uma nova chave sem expor os dados subjacentes, o ReEncrypt comando também pode ser usado.

> **Note**
>
> Ao usar as encrypt/decrypt funções, presume-se que todas as entradas estejam em hexBinary - por exemplo, um valor de 1 será inserido como 31 (hexadecimal) e um t minúsculo será representado como 74 (hexadecimal). Todas as saídas também serão geradas em hexBinary.

Para obter detalhes sobre todas as opções disponíveis, consulte o Guia de API para criptografar, descriptografare recriptografar.

Tópicos

- Criptografar dados
- Descriptografar dados

Criptografar dados

A `Encrypt Data` API é usada para criptografar dados usando chaves de criptografia de dados simétricas e assimétricas, bem como chaves derivadas de DUKPT e EMV. Vários algoritmos e variações são compatíveis, incluindo `TDES`, `RSA` e `AES`.

As entradas primárias são a chave de criptografia usada para criptografar os dados, os dados de texto simples no formato HexBinary a serem criptografados e os atributos de criptografia, como vetor e modo de inicialização, para cifras de bloco, como TDES. Os dados em texto simples precisam estar em múltiplos de 8 bytes para`TDES`, 16 bytes para `AES` e o tamanho da chave no caso de. `RSA` As entradas de chave simétricas (TDES, AES, DUKPT, EMV) devem ser preenchidas nos casos em que os dados de entrada não atendam a esses requisitos. A tabela a seguir mostra o tamanho máximo do texto sem formatação para cada tipo de chave e o tipo de preenchimento que você define `EncryptionAttributes` para as chaves RSA.

Tipo de preenchimento	RSA_2048	RSA_3072	RSA_4096
`OAEP SHA1`	428	684	940
`OAEP SHA256`	380	636	892

Tipo de preenchimento	RSA_2048	RSA_3072	RSA_4096
OAEP SHA512	252	508	764
PKCS1	488	744	1000
None	488	744	1000

As saídas primárias incluem os dados criptografados como texto cifrado no formato hexBinary e o valor da soma de verificação da chave de criptografia. Para obter detalhes sobre todas as opções disponíveis, consulte o Guia de API do Encrypt.

Exemplos

- Criptografe dados usando a chave simétrica AES
- Criptografe dados usando a chave DUKPT
- Criptografe dados usando a chave simétrica derivada do EMV
- Criptografe dados usando uma chave RSA

Criptografe dados usando a chave simétrica AES

> **Note**
>
> Todos os exemplos presumem que a chave relevante já existe. As chaves podem ser criadas usando a CreateKeyoperação ou importadas usando a ImportKeyoperação.

Example

Neste exemplo, criptografaremos dados em texto simples usando uma chave simétrica que foi criada usando a CreateKeyOperação ou importada usando a Operação. ImportKey Para essa operação, a chave deve ter sido KeyModesOfUse definida como `Encrypt` e KeyUsage definida como`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Consulte Chaves para operações criptográficas para obter mais opções.

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

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

Criptografe dados usando a chave DUKPT

Example

Neste exemplo, criptografaremos dados em texto simples usando uma chave DUKPT. AWS Suportes de criptografia de pagamento `TDES` e chaves `AES` DUKPT. Para essa operação, a chave deve ter sido KeyModesOfUse definida como `DeriveKey` e KeyUsage definida como`TR31_B0_BASE_DERIVATION_KEY`. Consulte Chaves para operações criptográficas para obter mais opções.

```
$ 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"
}
```

Criptografe dados usando a chave simétrica derivada do EMV

Example

Neste exemplo, criptografaremos dados de texto não criptografado usando uma chave simétrica derivada do EMV que já foi criada. Você pode usar um comando como esse para enviar dados para um cartão EMV. Para essa operação, a chave deve ter sido KeyModesOfUse definida como `Derive` e KeyUsage definida como `TR31_E1_EMV_MKEY_CONFIDENTIALITY` ou`TR31_E6_EMV_MKEY_OTHER`. Consulte Chaves para operações criptográficas para obter mais detalhes.

```
$ 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"
}
```

Criptografe dados usando uma chave RSA

Example

Neste exemplo, criptografaremos dados em texto simples usando uma chave pública RSA que foi importada usando a operação. ImportKey Para essa operação, a chave deve ter sido KeyModesOfUse definida como `Encrypt` e KeyUsage definida como`TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`. Consulte Chaves para operações criptográficas para obter mais opções.

Para esquemas de preenchimento como o PKCS #7 ou outros atualmente não compatíveis, aplique antes de chamar o serviço e selecione nenhum preenchimento ao omitir o indicador de preenchimento '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"
}
```

Descriptografar dados

A `Decrypt Data` API é usada para descriptografar dados usando chaves de criptografia de dados simétricas e assimétricas, bem como chaves derivadas de DUKPT e EMV. Vários algoritmos e variações são compatíveis, incluindo `TDES`, `RSA` e `AES`.

As entradas primárias são a chave de descriptografia usada para descriptografar dados, dados de texto cifrado no formato hexBinary a serem descriptografados e atributos de descriptografia, como vetor de inicialização, modo como cifras de bloco etc. As saídas primárias incluem os dados descriptografados como texto simples no formato hexBinary e o valor da soma de verificação da chave de decodificação. Para obter detalhes sobre todas as opções disponíveis, consulte o Guia de API para desencriptação.

Exemplos

- Descriptografe dados usando a chave simétrica AES
- Descriptografe dados usando a chave DUKPT
- Descriptografe dados usando a chave simétrica derivada do EMV
- Descriptografe dados usando uma chave RSA

Descriptografe dados usando a chave simétrica AES

Example

Neste exemplo, decifraremos dados de texto cifrado usando uma chave simétrica. Este exemplo mostra uma `AES` chave, mas `TDES_2KEY` ela também `TDES_3KEY` é suportada. Para essa operação, a chave deve ter sido KeyModesOfUse definida como `Decrypt` e KeyUsage definida como`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Consulte Chaves para operações criptográficas para obter mais opções.

```
$ 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"
}
```

Descriptografe dados usando a chave DUKPT

> **Note**
>
> O uso de dados de decodificação com DUKPT para transações P2PE pode retornar o PAN do cartão de crédito e outros dados do titular do cartão ao seu aplicativo, que precisarão ser contabilizados ao determinar o escopo do PCI DSS.

Example

Neste exemplo, decifraremos dados de texto cifrado usando uma chave DUKPT que foi criada usando a Operação ou importada usando a CreateKeyOperação. ImportKey Para essa operação, a chave deve ter sido KeyModesOfUse definida como `DeriveKey` e KeyUsage definida como`TR31_B0_BASE_DERIVATION_KEY`. Consulte Chaves para operações criptográficas para obter mais opções. Quando você usa `DUKPT`, para o algoritmo `TDES`, o comprimento dos dados do texto cifrado deve ser um múltiplo de 16 bytes. Para o algoritmo `AES`, o comprimento dos dados do texto cifrado deve ser um múltiplo de 32 bytes.

```
$ 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"
}
```

Descriptografe dados usando a chave simétrica derivada do EMV

Example

Neste exemplo, decifraremos dados de texto cifrado usando uma chave simétrica derivada do EMV que foi criada usando a operação ou importada usando a operação. CreateKeyImportKey Para essa operação, a chave deve ter sido KeyModesOfUse definida como `Derive` e KeyUsage definida como `TR31_E1_EMV_MKEY_CONFIDENTIALITY` ou`TR31_E6_EMV_MKEY_OTHER`. Consulte Chaves para operações criptográficas para obter mais detalhes.

```
$ 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": "31323334313233343132333431323334"
}
```

Descriptografe dados usando uma chave RSA

Example

Neste exemplo, decifraremos dados de texto cifrado usando um par de chaves RSA que foi criado usando a operação. CreateKey Para essa operação, a chave deve estar KeyModesOfUse configurada para habilitar `Decrypt` e KeyUsage definida como`TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`. Consulte Chaves para operações criptográficas para obter mais opções.

Para esquemas de preenchimento como o PKCS #7 ou outros atualmente não compatíveis, selecione nenhum preenchimento ao omitir o indicador de preenchimento 'Asymmetric= {}' e remova o preenchimento após chamar o serviço.

```
$ 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": "313233343132333431323334313233343"
}
```

Gerar e verificar dados do cartão

Gere e verifique os dados do cartão incorpora dados derivados dos dados do cartão, por exemplo, CVV CVV2, CVC e DCVV.

Tópicos

- Gerar dados do cartão
- Verificar dados do cartão

Gerar dados do cartão

A `Generate Card Data` API é usada para gerar dados do cartão usando algoritmos como CVV CVV2 ou Dynamic CVV2. Para saber quais chaves podem ser usadas para esse comando, consulte a seção Chaves válidas para operações criptográficas.

Muitos valores criptográficos, como CVV, iCVV CVV2, CAVV V7, usam o mesmo algoritmo criptográfico, mas variam os valores de entrada. Por exemplo, CardVerificationValue1 tem entradas de ServiceCode, Número do cartão e data de validade. Embora CardVerificationValue2 tenha apenas duas dessas entradas, isso ocorre porque para CVV2/CVC2, o ServiceCode é fixado em 000. Da mesma forma, para iCVV, o ServiceCode é fixado em 999. Alguns algoritmos podem reutilizar os campos existentes, como CAVV V8. Nesse caso, você precisará consultar o manual do provedor para obter os valores de entrada corretos.

> **Note**
>
> A data de expiração deve ser inserida no mesmo formato (como MMYY versus YYMM) para que a geração e a validação produzam resultados corretos.

Gerar CVV2

Example

Neste exemplo, geraremos um CVV2 para um determinado PAN com entradas PAN e data de validade do cartão. Isso pressupõe que você tenha gerado uma chave de verificação do cartão.

```
$ 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"
  }
```

Gerar iCVV

Example

Neste exemplo, geraremos um iCVV para um determinado PAN com entradas dePAN, um código de serviço de 999 e a data de validade do cartão. Isso pressupõe que você tenha gerado uma chave de verificação do cartão.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 no guia de referência da 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"
}
```

Verificar dados do cartão

O `Verify Card Data` é usado para verificar dados que foram criados usando algoritmos de pagamento que dependem de entidades principais de criptografia, como `DISCOVER_DYNAMIC_CARD_VERIFICATION_CODE`.

Os valores de entrada são normalmente fornecidos como parte de uma transação de entrada para um emissor ou parceiro de plataforma compatível. Para verificar um criptograma ARQC (usado para cartões com chips EMV), consulte Verificar ARQC.

Para obter mais informações, consulte VerifyCardValidationDatao guia da API.

Se o valor for verificado, a API retornará http/200. Se o valor não for verificado, ela retornará http/400.

Verificar CVV2

Example

Neste exemplo, validaremos um CVV/ CVV2 para um determinado PAN. Normalmente, CVV2 é fornecido pelo titular do cartão ou usuário durante o período da transação para validação. Para validar sua entrada, os seguintes valores serão fornecidos em tempo de execução - Chave de uso para validação (CVK)PAN, data de validade do cartão e CVV2 inseridos. O formato de vencimento do cartão deve corresponder ao utilizado na geração inicial do valor.

Para todos os parâmetros disponíveis, consulte CardVerificationValue2 no guia de referência da 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"
  }
```

Verifique o iCVV

Example

Neste exemplo, verificaremos um iCVV para um determinado PAN com entradas de Chave a ser usada para validação (CVK), um código de serviço de 999PAN, data de validade do cartão e o iCVV fornecido pela transação para validação.

iCVV não é um valor inserido pelo usuário (como CVV2), mas incorporado em um cartão EMV. Deve-se considerar se ele deve sempre ser validado quando fornecido.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 no guia de referência da 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"
}
```

Gerar, traduzir e verificar dados de PIN

As funções de dados de PIN permitem gerar PINs aleatórios, valores de verificação de PIN (PVV) e validar PINs criptografados de entrada em relação a compensações de PIN ou PVV.

A tradução de PINs permite traduzir um PIN de uma chave funcional para outra sem expor o PIN em texto não criptografado, conforme especificado pelo requisito n.º 1 do PCI PIN.

> **Note**
>
> Como a geração e validação de PINs geralmente são funções do emissor e a tradução de PINs é uma função típica do adquirente, recomendamos que você considere o acesso menos privilegiado e defina políticas apropriadas para o caso de uso do seu sistema.

Tópicos

- Traduzir dados de PIN
- Gerar dados de PIN
- Verificar dados de PIN

Traduzir dados de PIN

As funções de tradução de dados de PIN são usadas para traduzir dados de PIN criptografados de um conjunto de chaves para outro sem que os dados criptografados saiam do HSM. Isso é usado para criptografia P2PE, na qual as chaves de trabalho devem mudar, mas o sistema de processamento não precisa ou não tem permissão para descriptografar os dados. As entradas primárias são os dados criptografados, a chave de criptografia usada para criptografar os dados, os parâmetros usados para gerar os valores de entrada. O outro conjunto de entradas são os parâmetros de saída solicitados, como a chave a ser usada para criptografar e os parâmetros usados para criar essa saída. As saídas primárias são um conjunto de dados recém-criptografado, bem como os parâmetros usados para gerá-lo.

> **Note**
>
> Para conformidade com o PCI, os PrimaryAccountNumber valores de entrada e saída devem coincidir. Não é permitido traduzir um PIN de um PAN para outro.

Tópicos

- PIN de PEK para DUKPT
- PIN de PEK para PEK

PIN de PEK para DUKPT

Example

Neste exemplo, traduziremos um PIN de um bloco PIN AES ISO 4 usando a criptografia DUKPT para PEK TDES usando o bloco de PIN ISO 0. Isso é comum quando um terminal de pagamento criptografa um PIN na ISO 4 e, em seguida, ele pode ser traduzido de volta para o TDES para processamento posterior se a próxima conexão ainda não suportar 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"
        }
```

PIN de PEK para PEK

Example

Neste exemplo, traduzimos um PIN criptografado em uma PEK (chave de criptografia de PIN) para outra PEK. Isso geralmente é usado ao rotear transações entre diferentes sistemas ou parceiros que usam chaves de criptografia diferentes, mantendo a conformidade com o PIN PCI, mantendo o PIN criptografado durante todo o processo. Ambas as chaves usam criptografia TDES 3KEY neste exemplo, mas várias opções estão disponíveis, incluindo AES a TDES ISO-0, DUKPT ISO-4 a PEK ou AS2805 a PEK.

```
$ 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"
}
```

O bloco PIN de saída agora está criptografado sob o segundo PEK e pode ser transmitido com segurança para o sistema downstream que contém a chave correspondente.

Gerar dados de PIN

As funções de geração de dados de PIN são usadas para gerar PIN-related valores, como PVV e deslocamentos de blocos de pinos usados para validar a entrada de pinos pelos usuários durante o tempo de transação ou autorização. Essa API também pode gerar um novo PIN aleatório usando vários algoritmos.

Gere um pin aleatório e um Visa PVV correspondente

Example

Neste exemplo, geraremos um novo pino (aleatório) em que as saídas serão criptografadas `PIN block` (PinData.PinBlock) e PVV (pinoData.Offset). As principais entradas são `PAN`, `Pin Verification Key`, `Pin Encryption Key` e `PIN block format`.

Esse comando exige que a chave seja do tipo`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"
            }
        }
```

Gere um Visa PVV para um PIN conhecido

Example

Neste exemplo, geraremos um PVV para um determinado pino (criptografado). Um PIN criptografado pode ser recebido upstream, como de um terminal de pagamento ou de um titular de cartão, usando o fluxo de pinos selecionável pelo usuário. As principais entradas sãoPAN, o`Pin Verification Key`, o`Pin Encryption Key`, o `Encrypted Pin Block` e o. `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"
            }
        }
```

Gere o deslocamento de pinos IBM3624 para um pino

O IBM 3624 PIN Offset às vezes também é chamado de método IBM. Esse método gera um natural/intermediate PIN usando os dados de validação (normalmente o PAN) e uma chave PIN (PVK). Os pinos naturais são efetivamente um valor derivado e, sendo determinísticos, são muito eficientes para um emissor, pois nenhum dado de pino precisa ser armazenado no nível do titular do cartão. A desvantagem mais óbvia é que esse esquema não considera pinos aleatórios ou selecionáveis pelo titular do cartão. Para permitir esses tipos de pinos, um algoritmo de deslocamento foi adicionado ao esquema. O deslocamento representa a diferença entre o pino selecionado pelo usuário (ou aleatório) e a chave natural. O valor da compensação é armazenado pelo emissor do cartão ou pelo processador do cartão. No momento da transação, o serviço AWS de criptografia de pagamento

recalcula internamente o pino natural e aplica o deslocamento para encontrar o pino. Em seguida, ele compara isso com o valor fornecido pela autorização da transação.

Existem várias opções para o IBM3624:

- `Ibm3624NaturalPin`produzirá o pino natural e um bloco de pinos criptografado
- `Ibm3624PinFromOffset`gerará um bloco de pinos criptografado com um deslocamento
- `Ibm3624RandomPin`gerará um pino aleatório e, em seguida, o deslocamento correspondente e o bloco de pinos criptografado.
- `Ibm3624PinOffset`gera o deslocamento do pino dado um pino selecionado pelo usuário.

Internamente à criptografia de AWS pagamento, as seguintes etapas são executadas:

- Preencha o painel fornecido para 16 caracteres. Se forem fornecidos <16, coloque no lado direito usando o caractere de preenchimento fornecido.
- Criptografa os dados de validação usando a chave de geração de PIN.
- Decimalize os dados criptografados usando a tabela de decimalização. Isso mapeia dígitos hexidecimais em dígitos decimais, por exemplo, 'A' pode ser mapeado para 9 e 1 pode ser mapeado para 1.
- Obtenha os primeiros 4 dígitos de uma representação hexidecimal da saída. Esse é o alfinete natural.
- Se um pino selecionado pelo usuário ou for gerado aleatoriamente, o módulo subtrai o pino natural com o pin do cliente. O resultado é o deslocamento do pino.

Exemplos

- Exemplo: gerar deslocamento de pino IBM3624 para um pino

Exemplo: gerar deslocamento de pino IBM3624 para um pino

Neste exemplo, geraremos um novo pino (aleatório) em que as saídas serão um valor criptografado `PIN block` (PinData.PinBlock) e um valor de `IBM3624` deslocamento (pinoData.Offset). As entradas são `PAN` dados de validação (normalmente a panorâmica), caractere de preenchimento, o`Pin Verification Key`, o `Pin Encryption Key` e o. `PIN block format`

Esse comando exige que a chave de geração de pinos seja do tipo `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` e a chave de criptografia seja do tipo `TR31_P0_PIN_ENCRYPTION_KEY`

Example

O exemplo a seguir mostra a geração de um pino aleatório e, em seguida, a saída do bloco de pinos criptografado e do valor de deslocamento IBM3624 usando Ibm3624RandomPin

```
$ 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"
             }
         }
```

Verificar dados de PIN

As funções de verificação de dados de PIN são usadas para verificar se um PIN está correto. Isso normalmente envolve comparar o valor do PIN armazenado anteriormente com o que foi inserido pelo titular do cartão em um POI. Essas funções comparam dois valores sem expor o valor subjacente de nenhuma das fontes.

Valide o PIN criptografado usando o método PVV

Example

Neste exemplo, validaremos um PIN para um PAN específico. O PIN normalmente é fornecido pelo titular do cartão ou pelo usuário durante o período da transação para validação e é comparado com o valor registrado (a entrada do titular do cartão é fornecida como um valor criptografado pelo terminal ou outro provedor de upstream). Para validar essa entrada, os seguintes valores também serão fornecidos em tempo de execução: a chave usada para criptografar o pino de entrada (geralmente é um`IWK`) PAN e o valor a ser verificado (a ou). `PVV` `PIN offset`

Se a criptografia de AWS pagamento conseguir validar o PIN, um http/200 será devolvido. Se o pino não for validado, ele retornará um 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",

        }
```

Valide o PIN criptografado usando o método PVV - erro de pino incorreto

Example

Neste exemplo, tentaremos validar um PIN para um determinado PAN, mas falhará devido ao pino estar incorreto.

Ao usar SDKs, isso aparece como {"Message” :"Falha na verificação do bloqueio de pinos”. , "Razão” :"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.
```

Valide o PIN criptografado usando o método PVV - erro com entradas incorretas

Example

Neste exemplo, tentaremos validar um PIN para um determinado PAN, mas ele falhará devido a entradas incorretas e os dados recebidos não eram um PIN válido. As causas comuns são: os 2/ input parâmetros da 1/wrong chave que estão sendo usados, como formato de bloco panorâmico ou de pinos, estão incorretos, o 3/pin bloco está corrompido.

```
$ 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.
```

Valide um PIN em relação ao deslocamento de pinos IBM3624 armazenado anteriormente

Neste exemplo, você valida um PIN fornecido pelo titular do cartão em relação ao deslocamento do pino armazenado em arquivo com o emissor ou processador do cartão. As entradas incluem a chave de verificação (PVK), a chave de criptografia (PEK), o número da conta principal, o bloco de PIN criptografado do terminal de pagamento (ou de outro provedor upstream, como uma rede de cartões) e o deslocamento de pinos armazenado anteriormente. Se o PIN corresponder, a API retornará uma HTTP/200 resposta.

Esse comando exige que a chave de verificação do PIN seja do tipo `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` e a chave de criptografia seja do tipo`TR31_P0_PIN_ENCRYPTION_KEY`.

⚠ Validação do comprimento do PI

O `--pin-data-length` parâmetro opcional especifica o tamanho do PIN que está sendo verificado (4 a 12 dígitos). Se você não especificar esse parâmetro, o serviço valida todo o comprimento do PIN que o valor de compensação protege. Se você quiser validar somente uma parte do PIN, use o `--pin-data-length` parâmetro. Por exemplo, se `--pin-data-length` estiver definido como 4, mas o PIN real tiver 6 dígitos, o serviço valida somente os 4 dígitos mais à direita. Os 2 dígitos restantes não estão verificados. Você pode fazer isso intencionalmente se o PIN do titular do cartão tiver 6 dígitos, mas o PIN de entrada tiver apenas 4 dígitos, como quando a rede de entrada ou o comerciante suporta apenas um PIN de 4 dígitos.

Example

```
$ 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 \
    --encrypted-pin-block AC17DC148BDA645E \
    --pin-data-length 4 \
    --verification-attributes
 Ibm3624PinOffset="{DecimalizationTable=9876543210654321,PinValidationDataPadCharacter=D,PinVal
```

```
{
    "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"
}
```

Gere e verifique o ARQC

Um criptograma de solicitação de autorização (ARQC) é um criptograma gerado por um cartão com chip EMV (ou equivalente digital, como uma carteira móvel) durante uma transação de pagamento. O ARQC valida criptograficamente o cartão e garante a integridade dos dados da transação. Ele incorpora dados do cartão, dados do terminal e dados específicos da transação para produzir um valor exclusivo para cada transação.

AWS A criptografia de pagamento suporta duas operações ARQC. Você pode gerar um ARQC para fins de desenvolvimento e teste e pode verificar um ARQC para fluxos de autorização de transações de produção.

Tópicos

- Gerar criptograma de solicitação de autenticação (ARQC)
- Verificar o criptograma de solicitação de autenticação (ARQC)

Gerar criptograma de solicitação de autenticação (ARQC)

A API de geração de criptograma de solicitação de autenticação é usada para gerar ARQC. Essa API permite que você gere ARQC usando criptografia de AWS pagamento para fins de desenvolvimento e teste.

> **Important**
>
> Essa operação se destina somente a cenários de desenvolvimento e teste. Não é recomendável usar essa operação como um substituto para a geração de criptogramas com base em cartões nos fluxos de pagamento de produção.

Para ver todas as opções disponíveis, consulte GenerateAuthRequestCryptograma Referência da API.

Os criptogramas ARQC normalmente requerem as seguintes entradas (embora isso possa variar de acordo com a implementação):

- PAN - Especificado no PrimaryAccountNumber campo
- Número de sequência PAN (PSN) - especificado no campo PanSequenceNumber

- Método de derivação de chave, como Chave de sessão comum (CSK) - especificado no SessionKeyDerivationAttributes
- Modo de derivação de chave mestra (como EMV Opção A) - Especificado no MajorKeyDerivationMode
- Dados da transação - Uma sequência de dados da transação, do terminal e do cartão, como valor e data. Especificado no TransactionData campo.
- Chave mestra do emissor - A chave mestra usada para derivar a chave do criptograma (AC). Essa chave protege transações individuais e é especificada no KeyIdentifier campo.

Para obter mais informações sobre como criar e preencher dados de transações, consulte Criptograma de verificação de solicitação de autenticação (ARQC).

Tópicos

- Exemplos

Exemplos

Visa CVN10

Example

O exemplo a seguir gera um ARQC usando o Visa CVN10.

Se a criptografia AWS de pagamento gerar o ARQC com sucesso, um http/200 será retornado com o criptograma gerado.

```
$ aws payment-cryptography-data generate-auth-request-cryptogram \
--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",
  "AuthRequestCryptogram": "D791093C8A921769"
}
```

Visa CVN18 e Visa CVN22

Example

O exemplo a seguir gera um ARQC usando Visa CVN18 ou CVN22. As operações criptográficas são as mesmas entre CVN18 e CVN22, mas os dados contidos nos dados da transação variam. Comparado ao CVN10, um criptograma completamente diferente é gerado mesmo com as mesmas entradas.

Se a criptografia AWS de pagamento gerar o ARQC com sucesso, um http/200 será retornado com o criptograma gerado.

```
$ aws payment-cryptography-data generate-auth-request-cryptogram \
--key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
--major-key-derivation-mode EMV_OPTION_A \
--transaction-data
 00000000170000000000000008400080008000084016051700000000093800000B1F22010300000000000
 \
00000000000000000000000000000000000000000000800000000000000000 \
--session-key-derivation-attributes='{"EmvCommon":
{"ApplicationTransactionCounter":"000B", \
"PanSequenceNumber":"01","PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
  "KeyCheckValue": "08D7B4",
  "AuthRequestCryptogram": "61EDCC708B4C97B4"
}
```

Verificar o criptograma de solicitação de autenticação (ARQC)

A API do criptograma de solicitação de autenticação de verificação é usada para verificar o ARQC. A geração do ARQC normalmente é realizada em um cartão com chip EMV (ou equivalente digital, como carteira móvel) durante o período de autorização da transação, embora o serviço possa ser gerado para fins de desenvolvimento. Um ARQC é exclusivo para cada transação e tem como objetivo mostrar criptograficamente a validade do cartão e garantir que os dados da transação correspondam exatamente à transação atual (esperada).

AWS A criptografia de pagamento fornece uma variedade de opções para validar o ARQC e gerar valores ARPC opcionais, incluindo aqueles definidos no EMV 4.4 Livro 2 e outros esquemas usados

pela Visa e pela Mastercard. Para obter uma lista completa de todas as opções disponíveis, consulte a VerifyCardValidationData seção no Guia da API.

Os criptogramas ARQC normalmente requerem as seguintes entradas (embora isso possa variar de acordo com a implementação):

- PAN - Especificado no PrimaryAccountNumber campo
- Número de sequência PAN (PSN) - especificado no campo PanSequenceNumber
- Método de derivação de chave, como Chave de sessão comum (CSK) - especificado no SessionKeyDerivationAttributes
- Modo de derivação de chave mestra (como EMV Opção A) - Especificado no MajorKeyDerivationMode
- Dados da transação - uma sequência de vários dados de transação, terminal e cartão, como valor e data - especificados no TransactionData campo
- Chave mestra do emissor - a chave mestra usada para derivar a chave de criptograma (AC) usada para proteger transações individuais e especificada no campo KeyIdentifier

Tópicos

- Construir dados de transação
- Preenchimento de dados da transação
- Exemplos

Construir dados de transação

O conteúdo exato (e a ordem) do campo de dados da transação variam de acordo com a implementação e o esquema de rede, mas os campos mínimos recomendados (e a sequência de concatenação) são definidos no EMV 4.4 Livro 2, Seção 8.1.1 - Seleção de dados. Se os três primeiros campos forem valor (17,00), outro valor (0,00) e país de compra, isso resultaria nos dados da transação começando da seguinte forma:

- 000000001700: quantidade: 12 posições decimais implícitas de dois dígitos
- 000000000000: outro valor: 12 posições implícitas decimais de dois dígitos
- 0124: código de país de quatro dígitos
- Dados da transação de saída (parcial): 00000000170000000000000000124

Preenchimento de dados da transação

Os dados da transação devem ser preenchidos antes de serem enviados para o serviço. A maioria dos esquemas usa preenchimento ISO 9797 Método 2, em que uma string hexadecimal é anexada por hexadecimal 80 seguido por 00 até que o campo seja um múltiplo do tamanho do bloco de criptografia; 8 bytes ou 16 caracteres para TDES e 16 bytes ou 32 caracteres para AES. A alternativa (método 1) não é tão comum, mas usa somente 00 como caracteres de preenchimento.

Preenchimento ISO 9797 Método 1

Sem preenchimento:
0000000017000000000000000840008000800008401605170000000093800000B03011203 (74 caracteres ou 37 bytes)

Com preenchimento:
0000000017000000000000000840008000800008401605170000000093800000B03011203000000 (80 caracteres ou 40 bytes)

Preenchimento ISO 9797 Método 2

Sem preenchimento:
0000000017000000000000000840008000800008401605170000000093800000B1F220103000000 (80 caracteres ou 40 bytes)

Com preenchimento:
0000000017000000000000000840008000800008401605170000000093800000B1F220103000000800000 (88 caracteres ou 44 bytes)

Exemplos

Visa CVN10

Example

Neste exemplo, validaremos um ARQC gerado usando Visa CVN10.

Se a criptografia AWS de pagamento for capaz de validar o ARQC, um será devolvido. http/200 Se o ARCQ (Criptograma de Solicitação de Autorização) não for validado, ele retornará uma resposta. 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"
}
```

Visa CVN18 e Visa CVN22

Example

Neste exemplo, validaremos um ARQC gerado com o uso do Visa CVN18 ou CVN22. As operações criptográficas são as mesmas entre CVN18 e CVN22, mas os dados contidos nos dados da transação variam. Comparado ao CVN10, um criptograma completamente diferente é gerado mesmo com as mesmas entradas.

Se a criptografia AWS de pagamento for capaz de validar o ARQC, um será devolvido. http/200 Se o ARCQ não for validado, ele retornará um. 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"
}
```

Gerar e verificar MAC

Os Message Authentication Codes (MAC – códigos de autenticação de mensagens) são normalmente usados para autenticar a integridade de uma mensagem (independentemente de ela ter sido modificada). Hashes criptográficos, como HMAC (Código de Autenticação de Hash-Based Mensagem) CBC-MAC e CMAC (Código de Autenticação de Cipher-based Mensagem), fornecem garantia adicional ao remetente do MAC ao utilizar criptografia. O HMAC é baseado em funções de hash, enquanto o CMAC é baseado em cifras de bloco. O serviço também suporta os algoritmos ISO9797 1 e 3, que são tipos de. CBC-MACs

Todos os algoritmos MAC deste serviço combinam uma função hash criptográfica e uma chave secreta compartilhada. Eles usam uma mensagem e uma chave secreta, como o material de chave em uma chave, e retornam uma tag ou MAC exclusivo. Se até mesmo um caractere da mensagem mudar, ou se a chave secreta mudar, a tag resultante será totalmente diferente. Ao exigir uma chave secreta, os MACs criptográficos também fornecem autenticidade. É impossível gerar um MAC idêntico sem a chave secreta. Os MACs criptográficos às vezes são chamados de assinaturas simétricas porque funcionam como assinaturas digitais, mas usam uma única chave para assinatura e verificação.

AWS A criptografia de pagamento suporta vários tipos de MACs:

ALGORITMO 1 ISO9797

Indicado por `KeyUsage` ISO9797_ALGORITHM1. Se o campo não for um múltiplo do tamanho do bloco (8 caracteres bytes/16 hexadecimais para TDES, 16 bytes/32 caracteres para AES), a criptografia de AWS pagamento aplicará automaticamente o Método de preenchimento ISO9797 1. Se outros métodos de preenchimento forem necessários, você poderá aplicá-los antes de ligar para o serviço.

ALGORITMO 3 ISO9797 (MAC de varejo)

Indicado por `KeyUsage` ISO9797_ALGORITHM3. As mesmas regras de preenchimento se aplicam ao Algoritmo 1.

ALGORITMO 5 ISO9797 (CMAC)

Indicado por `KeyUsage` de TR31_M6_ISO_9797_5_CMAC_KEY

HMAC

Indicado por `KeyUsage` de TR31_M7_HMAC_KEY, incluindo HMAC_SHA224, HMAC_SHA256, HMAC_SHA384 e HMAC_SHA512

AS2805.4.1 MAC

Indicado por `KeyUsage` TR31_M0_ISO_16609_MAC_KEY. Para obter mais detalhes sobre o AS2805, consulte ???

MACARRÃO ESVAZIADO

O DUKPT MAC é normalmente usado para confirmar a origem e a carga útil dos terminais de pagamento de mensagens to/from . Ele deriva uma chave usando técnicas de derivação DUKPT

e, em seguida, executa o MAC. As chaves usadas com essa opção são indicadas por um de `KeyUsage` TR31_B0_BASE_DERIVATION_KEY.

EMV MAC

O EMV MAC é normalmente chamado de chave de integridade na documentação do EMV. Ele deriva uma chave usando técnicas de derivação EMV e, em seguida, utiliza ISO9797_ALGORITHM3 internamente. Normalmente é usado para enviar scripts do emissor para um cartão com chip para reprogramação. As chaves usadas com essa opção são indicadas por um de `KeyUsage` TR31_E2_EMV_MKEY_INTEGRITY. Se você estiver enviando um script e atualizando um PIN offline, verifique GenerateMacEmvPinChangese ele executa essas duas operações.

Tópicos

- Gerar MAC
- Verificar MAC

Gerar MAC

A API Generate MAC é usada para autenticar dados relacionados ao cartão, como rastrear dados de uma tarja magnética do cartão, usando chaves criptográficas conhecidas para gerar um MAC (Código de Autenticação de Mensagens) para validação de dados entre as partes emissoras e receptoras. Os dados usados para gerar MACs incluem dados de mensagens, chaves secretas de criptografia MAC e algoritmo MAC para gerar um valor MAC exclusivo para transmissão. A parte receptora do MAC usará os mesmos dados da mensagem MAC, chave de criptografia MAC e algoritmo para reproduzir outro valor MAC para comparação e autenticação de dados. Se qualquer caractere da mensagem for alterado ou a chave MAC usada para verificação não for idêntica, o valor MAC resultante será diferente. A API é compatível com o Algoritmo 1 do ISO 9797-1 e o Algoritmo 3 do ISO 9797-1 MAC (usando uma chave MAC estática e uma chave DUKPT derivada), chaves de criptografia HMAC e EMV MAC para essa operação.

O valor de entrada para `message-data` devem ser dados hexBinary.

Para obter mais informações sobre todas as opções dessa API, consulte GenerateMacVerifyMace.

O parâmetro opcional mac-length permite que você trunque o valor de saída (embora isso também possa ser feito dentro do seu código). Um comprimento de 8 se refere a 8 bytes ou 16 caracteres hexadecimais.

As chaves MAC podem ser criadas com criptografia AWS de pagamento por chamada CreateKeyou importadas por chamada ImportKey.

> **Note**
>
> Os algoritmos CMAC e HMAC não exigem preenchimento. Todos os outros exigem que os dados sejam preenchidos ao tamanho do bloco do algoritmo, que é múltiplo de 8 bytes (16 caracteres hexadecimais) para TDES e 16 bytes (32 caracteres hexadecimais) para AES.

Exemplos

- Gerar HMAC
- Gere MAC usando o algoritmo 3 do ISO 9797-1
- Gere MAC usando CMAC
- Gere MAC usando DUKPT CMAC

Gerar HMAC

Neste exemplo, geraremos um HMAC (Código de Autenticação de Hash-Based Mensagem) para autenticação de dados do cartão usando o algoritmo HMAC `HMAC_SHA256` e a chave de criptografia HMAC. A chave deve estar KeyUsage configurada como `TR31_M7_HMAC_KEY` e KeyModesOfUse para`Generate`. O comprimento do hash (por exemplo, 256) é definido quando a chave é criada e não pode ser modificada.

O parâmetro opcional mac-length reduzirá o MAC de saída, embora isso também possa ser executado fora do serviço. Esse valor está em bytes, portanto, um valor de 16 espera uma string hexadecimal de comprimento 32.

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"
}
```

Gere MAC usando o algoritmo 3 do ISO 9797-1

Neste exemplo, geraremos um MAC usando o Algoritmo 3 do ISO 9797-1 (MAC de varejo) para autenticação de dados do cartão. A chave deve estar KeyUsage configurada como `TR31_M3_ISO_9797_3_MAC_KEY` e KeyModesOfUse para`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"
}
```

Gere MAC usando CMAC

O CMAC é mais comumente usado quando as chaves são AES, mas também suporta TDES. Neste exemplo, geraremos um MAC usando o CMAC (ISO 9797-1 Algorithm 5) para autenticação de dados do cartão com uma chave AES. A chave deve estar KeyUsage configurada como `TR31_M6_ISO_9797_5_CMAC_KEY` e KeyModesOfUse para`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"
}
```

Gere MAC usando DUKPT CMAC

Este exemplo gera um MAC usando DUKPT (Derived Unique Key Per Transaction) com CMAC para autenticação de dados do cartão. A chave deve ter sido KeyUsage definida como verdadeira `TR31_B0_BASE_DERIVATION_KEY` e KeyModesOfUse `DeriveKey` definida como verdadeira.

As chaves DUKPT derivam uma chave exclusiva para cada transação usando uma chave de derivação básica (BDK) e um número de série da chave (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"
}
```

Verificar MAC

A API de verificação de MAC é usada para verificar o MAC (Código de autenticação de mensagens) para autenticação de dados relacionados a cartões. Ela deve usar a mesma chave de criptografia usada durante a geração do MAC para reproduzir o valor do MAC para autenticação. A chave de criptografia MAC pode ser criada com criptografia AWS de pagamento por chamada CreateKeyou importada por chamada ImportKey. A API suporta chaves de criptografia DUKPT MAC, HMAC e EMV MAC para essa operação.

Se o valor for verificado, o parâmetro de resposta `MacDataVerificationSuccessful` retornará `Http/200`, caso contrário, `Http/400` com uma mensagem indicando que `Mac verification failed`.

Exemplos

- Verifique o HMAC
- Verifique o MAC usando o DUKPT CMAC

Verifique o HMAC

Neste exemplo, verificaremos um HMAC (Código de Autenticação de Hash-Based Mensagem) para autenticação de dados do cartão usando o algoritmo HMAC `HMAC_SHA256` e a chave de criptografia HMAC. A chave deve ter sido KeyUsage definida como verdadeira `TR31_M7_HMAC_KEY` e KeyModesOfUse `Verify` definida como verdadeira.

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"
}
```

Verifique o MAC usando o DUKPT CMAC

Neste exemplo, verificaremos um MAC usando DUKPT (Derived Unique Key Per Transaction) com CMAC para autenticação de dados do cartão. A chave deve ter sido KeyUsage definida como verdadeira `TR31_B0_BASE_DERIVATION_KEY` e KeyModesOfUse `DeriveKey` definida como verdadeira. As chaves DUKPT derivam uma chave exclusiva para cada transação usando uma chave de derivação básica (BDK) e um número de série da chave (KSN). O valor de DukptKeyVariant deve corresponder entre o remetente e o destinatário. REQUEST normalmente será usado do terminal ao back-end, VERIFY do back-end ao terminal e BIDIRECTIONAL quando uma única chave é usada nas duas direções.

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"
}
```

Chaves válidas para operações criptográficas

Certas chaves podem ser usadas apenas para operações específicas. Além disso, algumas operações podem limitar os principais modos de uso das chaves. Consulte a tabela a seguir para ver as combinações permitidas.

> **Note**
>
> Certas combinações, embora permitidas, podem criar situações inutilizáveis, como gerar códigos CVV (`generate`), mas não conseguir verificá-los (`verify`).

Tópicos

- GenerateCardData
- VerifyCardData
- GeneratePinData (para VISA/ABA esquemas)
- GeneratePinData (para IBM3624)
- VerifyPinData (para VISA/ABA esquemas)
- VerifyPinData (para IBM3624)

- Descriptografar dados
- Criptografar dados
- Traduzir dados de PIN
- Generate/Verify MAC
- GenerateMacEmvPinChange
- VerifyAuthRequestCryptogram
- Import/Export Chave
- Tipos de chave não utilizados

GenerateCardData

Endpoint de API	Operação criptográfica ou algoritmo	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
GenerateCardData	• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	{ Generate = true },{ Generate = true, Verify = true }
GenerateCardData	• CARD_VERIFICATION_VALUE_1 • CARD_VERIFICATION_VALUE_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY	{ Generate = true },{ Generate = true, Verify = true }
GenerateCardData	• CARDHOLDER_AUTHENTICATION_V	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{ DeriveKey = verdadeiro}

Endpoint de API	Operação criptográfica ou algoritmo	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
	ERIFICATION_VALUE			
GenerateCardData	• DYNAMIC_CARD_VERIFICATION_CODE	TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS	• TDES_2KEY	{ DeriveKey = verdadeiro}
GenerateCardData	• DYNAMIC_CARD_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{ DeriveKey = verdadeiro}

VerifyCardData

Operação criptográfica ou algoritmo	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	{ Generate = true }, { Generate = true, Verify = true }
• CARD_VERIFICATION_VALUE_1	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KEY	{ Generate = true }, { Generate = true, Verify = true }

Operação criptográfica ou algoritmo	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
• CARD_VERIFICATION_VALUE_2			
• CARDHOLDER_AUTHENTICATION_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{ DeriveKey = verdadeiro}
• DYNAMIC_CARD_VERIFICATION_CODE	TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS	• TDES_2KEY	{ DeriveKey = verdadeiro}
• DYNAMIC_CARD_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{ DeriveKey = verdadeiro}

GeneratePinData (para VISA/ABA esquemas)

`VISA_PIN or VISA_PIN_VERIFICATION_VALUE`

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave de criptografia de PIN	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY	• { Encrypt = true, Wrap = true } • { Encrypt = true, Decrypt = true,

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
			Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Chave de geração de PIN	TR31_V2_VISA_PIN_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	• { Generate = true } • { Generate = true, Verify = true }

GeneratePinData (para **IBM3624)**

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave de criptografia de PIN	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY	Para IBM3624_NATURAL_PIN, IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET • { Encrypt = true, Wrap = true } • { Encrypt = true, Decrypt = true,

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
			Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro} Para IBM3624_PIN_OFFSET • { Encrypt = true, Unwrap = true } • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Chave de geração de PIN	TR31_V1_IBM3624_PIN_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	• { Generate = true } • { Generate = true, Verify = true }

VerifyPinData (para VISA/ABA esquemas)

`VISA_PIN`

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave de criptografia de PIN	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY	• { Decrypt = true, Unwrap = true } • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Chave de geração de PIN	TR31_V2_VISA_PIN_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	• { Verify = true } • { Generate = true, Verify = true }

VerifyPinData (para **IBM3624)**

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave de criptografia de PIN	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY	Para IBM3624_NATURAL_PIN, IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
			• { Decrypt = true, Unwrap = true } • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Chave de verificação de PIN	TR31_V1_IBM3624_PIN_VERIFICATION_KEY	• TDES_2KEY • TDES_3KEY	• { Verify = true } • { Generate = true, Verify = true }

Descriptografar dados

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = verdadeiro} • { NoRestrictions = verdadeiro}
EMV	TR31_E1_EMV_MKEY_CONFIDENTIALITY	• TDES_2KEY	• { DeriveKey = verdadeiro}

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
	TR31_E6_EMV_MKEY_OTHER		
RSA	TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION	• RSA_2048 • RSA_3072 • RSA_4096	• { Decrypt = true, Unwrap=true} • {Encrypt=true, Wrap=true,Decrypt = true, Unwrap=true}
Chaves simétricas	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Decrypt = true, Unwrap=true} • {Encrypt=true, Wrap=true,Decrypt = true, Unwrap=true} • { NoRestrictions = verdadeiro}

Criptografar dados

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = verdadeiro} • { NoRestrictions = verdadeiro}

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
EMV	TR31_E1_EMV_MKEY_CONFIDENTIALITY TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	• { DeriveKey = verdadeiro}
RSA	TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION	• RSA_2048 • RSA_3072 • RSA_4096	• { Encrypt = true, Wrap=true} • {Encrypt=true, Wrap=true,Decrypt = true, Unwrap=true}
Chaves simétricas	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Encrypt = true, Wrap=true} • {Encrypt=true, Wrap=true,Decrypt = true, Unwrap=true} • { NoRestrictions = verdadeiro}

Traduzir dados de PIN

Direction	Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Fonte de dados de entrada	DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = verdadeiro} • { NoRestrictions = verdadeiro}
Fonte de dados de entrada	Não-DUKPT (PEK, AWK, IWK etc)	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• { Decrypt = true, Unwrap = true } • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Destino de dados de saída	DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KEY • AES_128 • AES_192 • AES_256	• { DeriveKey = verdadeiro} • { NoRestrictions = verdadeiro}
Destino de dados de saída	Não DUKPT (PEK, IWK, AWK etc)	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• { Encrypt = true, Wrap = true } • { Encrypt = true, Decrypt

Direction	Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
				= true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}

Generate/Verify MAC

As chaves MAC são usadas para criar hashes criptográficos de um message/body dos dados. Não é recomendável criar uma chave com modos de uso de teclas limitados, pois você não poderá realizar a operação de correspondência. No entanto, você pode usar import/export uma chave com apenas uma operação se o outro sistema for destinado a executar a outra metade do par de operações.

Uso de chave permitido	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave MAC	TR31_M1_ISO_9797_1_MAC_KEY	• TDES_2KEY • TDES_3KEY	• { Generate = true } • { Generate = true, Verify = true } • { Verify = true } • { Generate = true }
Chave MAC (MAC de varejo)	TR31_M1_ISO_9797_3_MAC_KEY	• TDES_2KEY • TDES_3KEY	• { Generate = true } • { Generate = true, Verify = true } • { Verify = true } • { Generate = true }

Uso de chave permitido	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave MAC (CMAC)	TR31_M6_ISO_9797_5_CMAC_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• { Generate = true } • { Generate = true, Verify = true } • { Verify = true } • { Generate = true }
Chave MAC (HMAC)	TR31_M7_HMAC_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• { Generate = true } • { Generate = true, Verify = true } • { Verify = true }
Chave MAC (AS2805)	TR31_M0_ISO_16609_MAC_KEY	• TDES_2KEY • TDES_3KEY	• { Generate = true } • { Generate = true, Verify = true } • { Verify = true }

GenerateMacEmvPinChange

GenerateMacEmvPinChange combina geração de MAC e criptografia de PIN para operações de alteração de PIN offline do EMV. Essa operação requer dois tipos diferentes de chave: uma chave de integridade para geração de MAC e uma chave de confidencialidade para criptografia de PIN.

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
Chave de integrida de de mensagens seguras	TR31_E2_E MV_MKEY_I NTEGRITY	• TDES_2KEY	• { NoRestrictions = verdadeiro}
Chave de confidenc ialidade de mensagens seguras	TR31_E1_E MV_MKEY_C ONFIDENTIALITY	• TDES_2KEY	• { DeriveKey = verdadeiro}
PIN PEK (chave de criptografia PIN) atual	TR31_P0_P IN_ENCRYP TION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• { Decrypt = true, Unwrap = true } • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Novo PIN PEK (chave de criptografia de PIN)	TR31_P0_P IN_ENCRYP TION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• { Decrypt = true, Unwrap = true } • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true } • { NoRestrictions = verdadeiro}
Chave ARQC **Note** Só se aplica aos	TR31_E0_E MV_MKEY_A PP_CRYPTOGRAMS	• TDES_2KEY	• { DeriveKey = verdadeiro}

Tipo de chave	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
esquemas de derivação Visa e Amex.			

VerifyAuthRequestCryptogram

Uso de chave permitido	Opção de EMV	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
• OPÇÃO A • OPÇÃO B	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS	• TDES_2KEY	• { DeriveKey = verdadeiro}

Import/Export Chave

Tipo de operação	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
TR-31 Chave de embrulho	TR31_K1_KEY_BLOCK_PROTECTION_KEY TR31_K0_KEY_ENCRYPTION_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Encrypt = true, Wrap = true} (somente exportação) • {Decrypt = true, Unwrap = true}

Tipo de operação	Uso de chave permitido	Algoritmo de chave permitido	Combinação permitida dos principais modos de uso
			(somente importação) • { Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true }
Importação de CA confiável	TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE	• RSA_2048 • RSA_3072 • RSA_4096	• { Verify = true }
Importação de certificado de chave pública para criptografia assimétrica	TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION	• RSA_2048 • RSA_3072 • RSA_4096	• {Encrypt=true, wrap=true}
Chave usada para codificar algoritmos de acordo, como ECDH	TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT	• ECC_NIST_P256 • ECC_NIST_P384 • ECC_NIST_P521	• { DeriveKey = verdadeiro}

Tipos de chave não utilizados

Os seguintes tipos de chave não são usados atualmente pela criptografia AWS de pagamento

- TR31_P1_PIN_GENERATION_KEY

Casos de uso comuns

AWS A criptografia de pagamento suporta muitas operações criptográficas de pagamento típicas. Os tópicos a seguir servem como um guia sobre como usar essas operações para casos de uso comuns típicos. Para ver uma lista de todos os comandos, consulte a API AWS Payment Cryptography.

Tópicos

- Emissores e processadores de emissores
- Facilitadores de aquisição e pagamento

Emissores e processadores de emissores

Os casos de uso do emissor geralmente consistem em algumas partes. Esta seção é organizada por função (como trabalhar com pinos). Em um sistema de produção, as chaves normalmente têm como escopo um determinado compartimento de cartão e são criadas durante a configuração do compartimento, em vez de embutidas, conforme mostrado aqui.

Tópicos

- Funções gerais
- Funções específicas da rede

Funções gerais

Tópicos

- Gere um pino aleatório e o PVV associado e, em seguida, verifique o valor
- Gere ou verifique um CVV para um determinado cartão
- Gere ou verifique um CVV2 para um cartão específico
- Gere ou verifique um iCVV para um cartão específico
- Verifique um EMV ARQC e gere um ARPC
- Gere e verifique um MAC EMV
- Gere EMV MAC para alteração de PIN

Gere um pino aleatório e o PVV associado e, em seguida, verifique o valor

Tópicos

- Crie a (s) chave (s)
- Gere um pino aleatório, gere PVV e retorne o PIN e o PVV criptografados
- Valide o PIN criptografado usando o método PVV
- PINs gerenciados pelo usuário

Crie a (s) chave (s)

Para gerar um pino aleatório e o PVV, você precisará de duas chaves, uma chave de verificação de pinos (PVK) para gerar o PVV e uma chave de criptografia de pinos para criptografar o pino. O pino em si é gerado aleatoriamente com segurança dentro do serviço e não está relacionado criptograficamente a nenhuma das chaves.

O PGK deve ser uma chave do algoritmo TDES_2KEY com base no próprio algoritmo PVV. Um PEK pode ser TDES_2KEY, TDES_3KEY ou AES_128. Nesse caso, como o PEK é destinado ao uso interno em seu sistema, o AES_128 seria uma boa escolha. Se um PEK for usado para intercâmbio com outros sistemas (por exemplo, redes de cartões, adquirentes, caixas eletrônicos) ou estiver sendo movido como parte de uma migração, o TDES_2KEY pode ser a escolha mais apropriada por motivos de compatibilidade.

Crie o 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/ivi5ksfsuplneuyt Isso será necessário na próxima etapa.

Crie o 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
                }
            }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/ov6icy4ryas4zcza Isso será necessário na próxima etapa.

Gere um pino aleatório, gere PVV e retorne o PIN e o PVV criptografados

Example

Neste exemplo, geraremos um novo pino (aleatório) de 4 dígitos em que as saídas serão criptografadas `PIN block` (PinData.PinBlock) e um PVV (pinoData.VerificationValue). As principais entradas são `PAN` o `Pin Verification Key` (também conhecido como chave de geração de pinos), o `Pin Encryption Key` e o `formato de bloco de` PIN.

Esse comando exige que a chave seja do tipo`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"
            }
        }
```

Valide o PIN criptografado usando o método PVV

Example

Neste exemplo, validaremos um PIN para um PAN específico. O PIN normalmente é fornecido pelo titular do cartão ou pelo usuário durante o período da transação para validação e é comparado com o valor registrado (a entrada do titular do cartão é fornecida como um valor criptografado pelo terminal ou outro provedor de upstream). Para validar essa entrada, os seguintes valores também serão fornecidos em tempo de execução: o pino criptografado, a chave usada para criptografar o pino de entrada (geralmente chamado de IWK) PAN e o valor a ser verificado (um ou). PVV `PIN offset`

Se a criptografia de AWS pagamento conseguir validar o PIN, um http/200 será devolvido. Se o pino não for validado, ele retornará um 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",
}
```

PINs gerenciados pelo usuário

Tópicos

- Visão geral do
- Crie as chaves
- Definir um PIN selecionável pelo usuário
- Redefinir um PIN (PIN verde)
- Revelar um PIN

Visão geral do

AWS A criptografia de pagamento suporta fluxos de PIN gerenciados pelo usuário (gerenciados pelo cliente), nos quais um PIN é trocado com segurança entre o dispositivo do titular do cartão e o serviço por meio do back-end do cliente. O dispositivo e o serviço usam o acordo de chave de curva elíptica Diffie-Hellman (ECDH) para derivar uma chave simétrica compartilhada em cada lado, e essa chave protege um bloco de 4 PIN no formato ISO. ??? Como a chave é derivada em vez de trocada, o PIN claro e o bloco de PIN transparente nunca são transmitidos para fora do dispositivo.

Esses fluxos envolvem três partes. O dispositivo do titular do cartão gera seu par de chaves EC, deriva a chave ECDH AES-128 , criptografa ou descriptografa o bloco de 4 PIN no formato ISO e mantém o PIN claro; ele não tem credenciais da AWS. O back-end do cliente (seu emissor ou servidor processador) mantém as credenciais da AWS, liga para a criptografia de AWS pagamento, assina o certificado do dispositivo e armazena o PVV. ??? AWS A criptografia de pagamento executa as operações criptográficas. O dispositivo se comunica somente com o back-end do cliente, e o back-end do cliente faz todas as chamadas para a AWS criptografia de pagamento.

Este tópico aborda três fluxos:

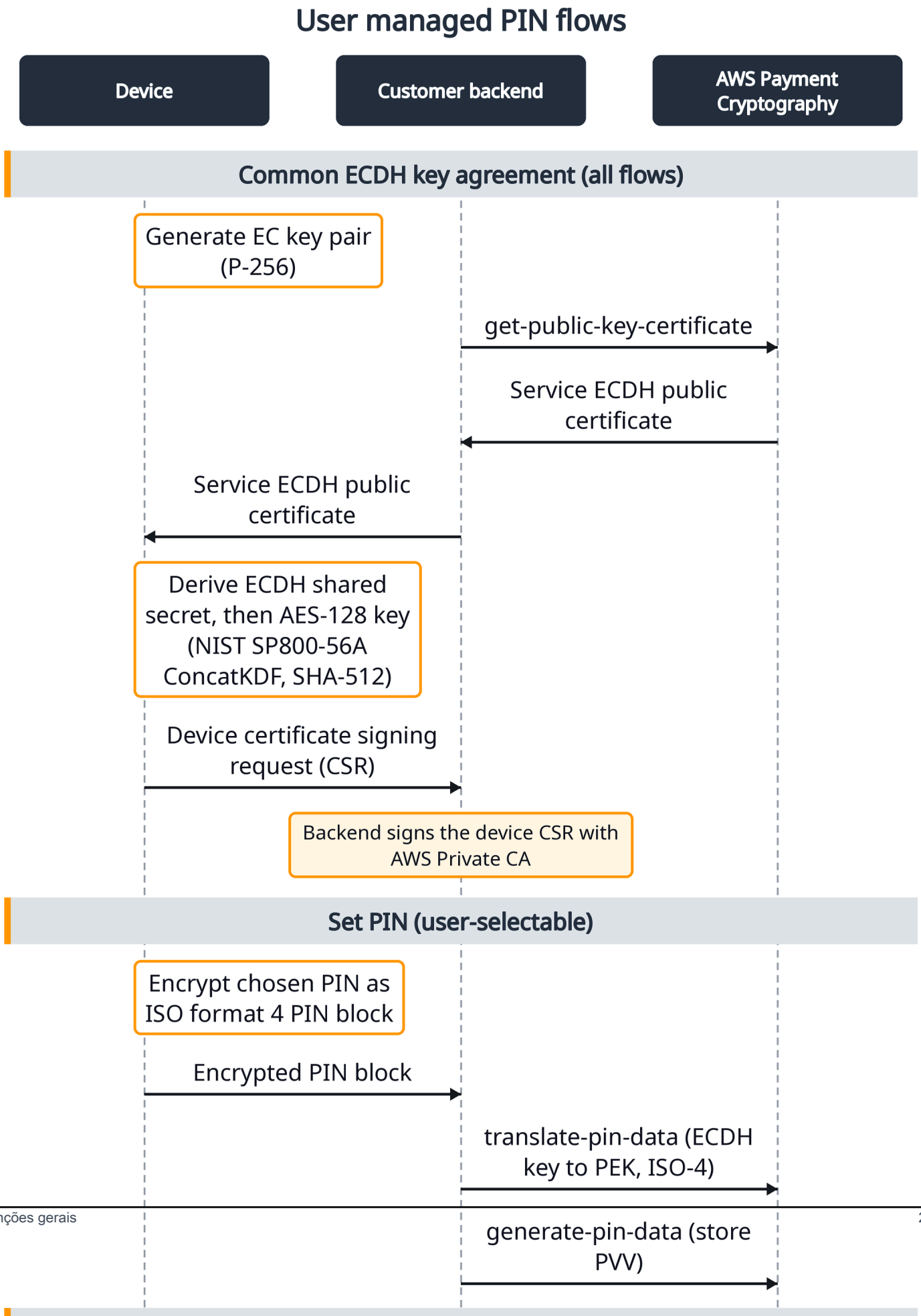
- User-selectable PIN (Definir PIN) - o titular do cartão escolhe seu próprio PIN no dispositivo.

- PIN verde (PIN de redefinição) - o serviço gera um PIN aleatório que é entregue ao titular do cartão, por exemplo, durante a primeira emissão ou uma redefinição de PIN esquecido.
- Revelar PIN - um bloco de PIN existente é devolvido a um dispositivo autorizado para casos de uso restritos.

Todos os três fluxos compartilham a mesma configuração de chave e a mesma mecânica de acordo de dispositivo ECDH, descrita em Criar as chaves. ??? Para obter detalhes sobre o contrato de chave em si, consulte Importar chaves usando técnicas assimétricas (ECDH).

> **Note**
>
> Esses fluxos lidam com PINs e blocos de PIN claros somente no dispositivo, nunca em trânsito para a criptografia AWS de pagamento. Analise cada fluxo cuidadosamente com sua equipe de segurança antes de implantá-lo.

O diagrama a seguir mostra os fluxos de PIN gerenciados pelo usuário.


User managed PIN flows
Device
Customer backend
AWS Payment Cryptography
Common ECDH key agreement (all flows)
Generate EC key pair (P-256)
get-public-key-certificate
Service ECDH public certificate
Service ECDH public certificate
Derive ECDH shared secret, then AES-128 key (NIST SP800-56A ConcatKDF, SHA-512)
Device certificate signing request (CSR)
Backend signs the device CSR with AWS Private CA
Set PIN (user-selectable)
Encrypt chosen PIN as ISO format 4 PIN block
Encrypted PIN block
translate-pin-data (ECDH key to PEK, ISO-4)
generate-pin-data (store PVV)

Para obter um exemplo completo e executável desses fluxos usando o AWS SDK para Python, consulte o conjunto de PIN do ECDH e a amostra de revelação de fluxos no repositório de amostras em. AWS GitHub

> **Note**
>
> O fluxo típico inclui um dispositivo titular de cartão que executa o contrato de chave ECDH e criptografa ou descriptografa o bloco de 4 PIN no formato ISO. Se você já protege os PINs entre o dispositivo e o back-end do cliente por outro meio, pode usar uma versão modificada desses fluxos diretamente entre o back-end do cliente e AWS a criptografia de pagamento. Nessa variante, o back-end do cliente realiza o contrato de chave ECDH e as operações de bloqueio de PIN no lugar do dispositivo.

Crie as chaves

Todos os três fluxos usam o mesmo conjunto de chaves: uma chave de contrato ECDH para o contrato de chave com o dispositivo, uma chave de pino (PGK) para gerar o PVV e uma Generation/Verification chave de criptografia de pinos (PEK) na qual o bloco de PIN é traduzido para armazenamento de back-end. Você também importa a chave pública da autoridade de certificação (CA) do dispositivo para que a criptografia AWS de pagamento confie nos certificados do dispositivo apresentados durante a tradução.

Crie a chave do contrato ECDH. Essa chave é usada para derivar a chave simétrica compartilhada com o dispositivo.

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

A resposta reflete os parâmetros da solicitação e inclui o ARN da chave ECDH.

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxxx",
        "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": "1A2B3C",
        "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"
    }
}
```

Crie a Generation/Verification chave de pinos (PGK) usada para gerar o PVV.

```
$ aws payment-cryptography create-key \
    --exportable \
    --key-attributes KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_V2_VISA_PIN_VERIFICATION_KEY,
\
        KeyClass=SYMMETRIC_KEY,\
        KeyModesOfUse='{Generate=true,Verify=true}'
```

```
{
    "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"
    }
}
```

Crie a chave de criptografia de pinos (PEK) na qual o bloco de PIN do dispositivo é traduzido para armazenamento de back-end.

```
$ 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}'
```

```
{
    "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": true,
                "Decrypt": true,
```

```
                "Wrap": true,
                "Unwrap": true,
                "Generate": false,
                "Sign": false,
                "Verify": false,
                "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"
    }
}
```

Importe a chave pública da CA do dispositivo para que a criptografia de AWS pagamento possa verificar os certificados do dispositivo apresentados em chamadas de tradução posteriores. <base64-CA-cert>Substitua pelo certificado CA codificado em base64.

```
$ aws payment-cryptography import-key \
    --key-material='{"RootCertificatePublicKey":{"KeyAttributes":
{"KeyAlgorithm":"ECC_NIST_P256","KeyClass":"PUBLIC_KEY","KeyModesOfUse":
{"Verify":true},"KeyUsage":"TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"},"PublicKeyCertificat
CA-cert>"}}'
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "ECC_NIST_P256",
            "KeyModesOfUse": {
                "Verify": true
            }
        },
```

```
        "KeyCheckValue": "4D5E6F",
        "KeyCheckValueAlgorithm": "CMAC",
        "Enabled": true,
        "Exportable": false,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
        "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
    }
}
```

Anote cada devolvidoKeyArn. Os exemplos a seguir usam os seguintes espaços reservados literais:

- Chave do acordo ECDH: `arn:aws:payment-cryptography:us-east-1:111122223333:key/xxxxxxxxxxxxxxxx`
- Chave pública CA importada: `arn:aws:payment-cryptography:us-east-1:111122223333:key/yyyyyyyyyyyyyyyy`
- PGK: `arn:aws:payment-cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza`
- PEK: `arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt`

No dispositivo do titular do cartão (normalmente um telefone celular ou dispositivo similar), as seguintes etapas são executadas no código do aplicativo do dispositivo para cada um dos fluxos Definir PIN, Redefinir PIN e Revelar PIN: gerar um par de chaves EC na curva P-256; obter o certificado público ECDH de serviço do back-end do cliente; derivar o segredo compartilhado do ECDH e, em seguida, uma AES-128 chave de 16 bytes usando o NIST SP800-56A ConcatKDF (SHA-5 12) com o `SharedInformation` valor; e criptografar ou descriptografar a ISO formate um bloco de 4 PIN com essa chave derivada. ??? O dispositivo não tem credenciais da AWS e não liga diretamente para a criptografia de AWS pagamento; ele troca esses dados com o back-end do cliente, que faz as chamadas de serviço.

Como o dispositivo geralmente é um telefone, seu par de chaves EC e o certificado são provisionados na primeira utilização e atualizados periodicamente a partir de então. O dispositivo gera o par de chaves EC e uma solicitação de assinatura de certificado (CSR), e o back-end do cliente assina essa CSR usando uma CA CA privada da AWS ou similar. O back-end do cliente usa o certificado de dispositivo assinado resultante em suas chamadas de translate-pin-data, e a criptografia de AWS pagamento o verifica em relação à chave pública da CA que você importou.

O back-end do cliente obtém o certificado público ECDH do serviço e o fornece ao dispositivo. O certificado é retornado como um PEM codificado em base64 no `KeyCertificate` campo, que é decodificado em base64 em um arquivo PEM. O dispositivo usa esse certificado como chave pública do serviço quando deriva o segredo compartilhado.

```
$ aws payment-cryptography get-public-key-certificate \
    --key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx
```

> **Note**
>
> `SharedInformation`são 32 bytes aleatórios gerados no dispositivo e passados pelo back-end do cliente para a criptografia de AWS pagamento como uma string hexadecimal. Ele deve corresponder ao valor usado pelo dispositivo no KDF, ou as chaves derivadas não concordarão.

Definir um PIN selecionável pelo usuário

No fluxo Definir PIN, o titular do cartão escolhe seu próprio PIN no dispositivo (por exemplo,`1234`). O dispositivo obtém a chave compartilhada ECDH, criptografa o PIN escolhido em um bloco de 4 PIN no formato ISO sob essa chave e envia o bloco PIN criptografado, sua solicitação de assinatura de certificado (CSR) e as informações compartilhadas para o back-end do cliente. O back-end do cliente assina a CSR e, em seguida, realiza duas chamadas no plano de dados. Este exemplo usa a amostra PAN`171234567890123`. Os valores do bloco PIN, do certificado e das informações compartilhadas mostrados são espaços reservados fictícios.

Primeiro, traduza o bloco ECDH-protected PIN para o PEK do servidor. A chave de entrada é a chave do contrato ECDH, empacotada com Diffie-Hellman os parâmetros e o certificado do dispositivo assinado; a chave de saída é a PEK.

```
$ aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block A1B2C3D4E5F60718 \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx \
    --incoming-wrapped-key
 'WrappedKeyMaterial={DiffieHellmanSymmetricKey={CertificateAuthorityPublicKeyIdentifier=arn:aw
cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy,KeyAlgorithm=AES_128,KeyDerivationFunction=NIST_SP800,KeyDerivationHashAlgorit
signed-device-
```

```
cert>,SharedInformation=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef}}'
 \
    --incoming-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --outgoing-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}'
```

A resposta contém o bloco PIN recriptografado sob o PEK no formato ISO 4.

```
{
    "PinBlock": "F7E6D5C4B3A29180",
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "KeyCheckValue": "7CC9E2"
}
```

Em seguida, gere o PVV para o bloco PEK-encrypted PIN. O PGK gera o PVV e o PEK identifica a chave sob a qual o bloco PIN fornecido está criptografado.

```
$ aws payment-cryptography-data generate-pin-data \
    --generation-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza \
    --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --pin-block-format ISO_FORMAT_4 \
    --primary-account-number 171234567890123 \
    --generation-attributes
 'VisaPinVerificationValue={PinVerificationKeyIndex=1,EncryptedPinBlock=F7E6D5C4B3A29180}'
```

A resposta contém a entrada PVV `PinData.VerificationValue` e o bloco PIN criptografado. Armazene o PVV no back-end do cliente para que você possa verificar o PIN posteriormente. Para verificação, consulte Gerar um pino aleatório e o PVV associado e, em seguida, verifique o valor.

```
{
    "GenerationKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
    "GenerationKeyCheckValue": "51A200",
    "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "EncryptionKeyCheckValue": "7CC9E2",
```

```
    "EncryptedPinBlock": "F7E6D5C4B3A29180",
    "PinData": {
        "VerificationValue": "5507"
    }
}
```

Para obter mais informações, consulte TranslatePinData e GeneratePinData no guia de referência da API.

Redefinir um PIN (PIN verde)

No fluxo de redefinição de PIN (PIN verde), a criptografia de AWS pagamento gera um PIN aleatório que o back-end do cliente retorna ao dispositivo para exibir ao titular do cartão, por exemplo, durante a primeira emissão ou uma redefinição de PIN esquecido. O dispositivo executa a mesma configuração de ECDH descrita em Criar as chaves, e o back-end do cliente faz as chamadas do plano de dados em seu nome. Este exemplo usa a amostra PAN`171234567890123`; o bloco de PIN e os valores das informações compartilhadas são espaços reservados fictícios.

Primeiro, gere um PIN aleatório e seu PVV, criptografando o bloco de PIN sob o PEK.

```
$ aws payment-cryptography-data generate-pin-data \
    --generation-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza \
    --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --pin-block-format ISO_FORMAT_4 \
    --primary-account-number 171234567890123 \
    --generation-attributes 'VisaPin={PinVerificationKeyIndex=1}'
```

A resposta contém a entrada PVV `PinData.VerificationValue` e o bloco PEK-encrypted PIN (formato ISO 4). Armazene o PVV no back-end do cliente para verificação posterior.

```
{
    "GenerationKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
    "GenerationKeyCheckValue": "51A200",
    "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "EncryptionKeyCheckValue": "7CC9E2",
    "EncryptedPinBlock": "F7E6D5C4B3A29180",
    "PinData": {
        "VerificationValue": "5507"
```

```
    }
}
```

Em seguida, traduza o bloco PEK-encrypted PIN (formato ISO 4) na ECDH-derived chave do dispositivo (formato ISO 4) para que somente o dispositivo solicitante possa lê-lo.

```
$ aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block F7E6D5C4B3A29180 \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --incoming-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx \
    --outgoing-wrapped-key
 'WrappedKeyMaterial={DiffieHellmanSymmetricKey={CertificateAuthorityPublicKeyIdentifier=arn:aw
cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy,KeyAlgorithm=AES_128,KeyDerivationFunction=NIST_SP800,KeyDerivationHashAlgorit
signed-device-
cert>,SharedInformation=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef}}'
 \
    --outgoing-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}'
```

A resposta contém o bloco PIN recriptografado sob a ECDH-derived chave do dispositivo no formato ISO 4.

```
{
    "PinBlock": "A1B2C3D4E5F60718",
    "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx",
    "KeyCheckValue": "1A2B3C"
}
```

O back-end do cliente retorna o bloco de ECDH-encrypted PIN para o dispositivo. O dispositivo decifra o bloco de PIN devolvido com sua ECDH-derived chave (no código do aplicativo do dispositivo) e exibe o PIN para o titular do cartão. Como a chave é derivada somente desse dispositivo, nenhuma outra parte pode descriptografar o bloco de PIN.

Para obter mais informações, consulte GeneratePinData e TranslatePinData no guia de referência da API.

Revelar um PIN

O fluxo Revelar PIN retorna um bloco de PIN existente para o dispositivo do titular do cartão para que o dispositivo possa exibir ou revelar o PIN. O dispositivo executa a mesma configuração de ECDH descrita em Criar as chaves, e o back-end do cliente traduz o bloco de PEK-encrypted PIN armazenado de volta para a ECDH-derived chave e o devolve ao dispositivo. Este exemplo usa a amostra PAN`171234567890123`; o bloco de PIN e os valores das informações compartilhadas são espaços reservados fictícios.

> **Note**
>
> Considere autenticar o titular do cartão e permitir que ele defina um novo PIN em vez de revelar o PIN existente. Uma exceção notável é em países que usam PIN offline, por exemplo, na Europa, onde o PIN também é armazenado no cartão com chip. Nesses casos, revelar o PIN existente pode ser mais simples do que reprogramar o cartão.

Traduza o bloco PEK-encrypted PIN armazenado (formato ISO 4) na ECDH-derived chave do dispositivo (formato ISO 4).

```
$ aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block F7E6D5C4B3A29180 \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --incoming-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx \
    --outgoing-wrapped-key
 'WrappedKeyMaterial={DiffieHellmanSymmetricKey={CertificateAuthorityPublicKeyIdentifier=arn:aw
cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy,KeyAlgorithm=AES_128,KeyDerivationFunction=NIST_SP800,KeyDerivationHashAlgorit
signed-device-
cert>,SharedInformation=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef}}'
 \
    --outgoing-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}'
```

A resposta contém o bloco PIN recriptografado sob a ECDH-derived chave do dispositivo no formato ISO 4.

```
{
    "PinBlock": "A1B2C3D4E5F60718",
    "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx",
    "KeyCheckValue": "1A2B3C"
}
```

O back-end do cliente retorna o bloco de ECDH-encrypted PIN para o dispositivo. O dispositivo decifra o bloco de PIN retornado com sua ECDH-derived chave (no código do aplicativo do dispositivo) para revelar o PIN. Como a chave é derivada somente desse dispositivo, nenhuma outra parte pode descriptografar o bloco de PIN.

Para obter mais informações, consulte TranslatePinData o guia de referência da API.

Gere ou verifique um CVV para um determinado cartão

CVV ou CVV1 é um valor tradicionalmente incorporado na faixa magnética de um cartão. Não é o mesmo que o CVV2 (visível para o titular do cartão e para uso em compras on-line).

A primeira etapa é criar uma chave. Para este tutorial, você cria uma chave CVK 3DES (2KEY TDES) de comprimento duplo.

> **Note**
>
> CVV, CVV2 e iCVV usam algoritmos semelhantes, se não idênticos, mas variam os dados de entrada. Todos usam o mesmo tipo de chave TR31_C0_CARD_VERIFICATION_KEY, mas é recomendável usar chaves separadas para cada finalidade. Eles podem ser diferenciados usando and/or tags de aliases, como no exemplo abaixo.

Crie a chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/r52o3wbqxyf6qlqr Isso será necessário na próxima etapa.

Gere um CVV

Example

Neste exemplo, geraremos um CVV para um determinado PAN com entradas dePAN, um código de serviço (conforme definido pelo ISO/IEC 7813) de 121 e a data de validade do cartão.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 o guia de referência da 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"
                }
```

Validar CVV

Example

Neste exemplo, verificaremos um CVV para um determinado PAN com entradas de um CVK, um código de serviço 121PAN, a data de validade do cartão e o CVV fornecido durante a transação para validação.

Para ver todos os parâmetros disponíveis, CardVerificationValue1 consulte o guia de referência da API.

> **Note**
>
> CVV não é um valor inserido pelo usuário (como CVV2), mas normalmente está incorporado em uma tarja magnética. Deve-se considerar se ele deve sempre ser validado quando fornecido.

```
$ 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"
}
```

Gere ou verifique um CVV2 para um cartão específico

O CVV2 é um valor tradicionalmente fornecido no verso de um cartão e usado para compras on-line. Para cartões virtuais, ele também pode ser exibido em um aplicativo ou em uma tela. Criptograficamente, é igual ao CVV1, mas com um valor de código de serviço diferente.

Crie a chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/7f7g4spf3xcklhzu Isso será necessário na próxima etapa.

Gere um CVV2

Example

Neste exemplo, geraremos um CVV2 para um determinado PAN com entradas PAN e data de validade do cartão.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue2 o guia de referência da 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"
                    }
```

Validar um CVV2

Example

Neste exemplo, verificaremos um CVV2 para um determinado PAN com entradas de um CVK, a data de validade do cartão PAN e o CVV fornecido durante a transação para validação.

Para ver todos os parâmetros disponíveis, CardVerificationValue2 consulte o guia de referência da API.

> **Note**
>
> CVV2 e as outras entradas são valores inseridos pelo usuário. Dessa forma, não é necessariamente um sinal de um problema que isso periodicamente não seja validado.

```
$ 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"
   }
```

Gere ou verifique um iCVV para um cartão específico

O iCVV usa o mesmo algoritmo, CVV/CVV2 mas o iCVV está incorporado em um cartão com chip. Seu código de serviço é 999.

Crie a chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
               }
           }
```


Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/c7dsi763r6s7lfp3 Isso será necessário na próxima etapa.

Gere um iCVV

Example

Neste exemplo, geraremos um iCVV para um determinado PAN com entradas dePAN, um código de serviço (conforme definido por ISO/IEC 7813) de 999 e a data de validade do cartão.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 o guia de referência da 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"
                    }
```

Validar o iCVV

Example

Para validação, as entradas são CVK, código de serviço 999PAN, data de validade do cartão e o iCVV fornecido durante a transação para validação.

Para ver todos os parâmetros disponíveis, CardVerificationValue1 consulte o guia de referência da API.

> **Note**
>
> O iCVV não é um valor inserido pelo usuário (como CVV2), mas normalmente está incorporado em um cartão. EMV/chip Deve-se considerar se ele deve sempre ser validado quando fornecido.

```
$ 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"
   }
```

Verifique um EMV ARQC e gere um ARPC

O ARQC (Authorization Request Cryptogram) é um criptograma gerado por um cartão EMV (chip) e usado para validar os detalhes da transação, bem como o uso de um cartão autorizado. Ele incorpora dados do cartão, do terminal e da própria transação.

No momento da validação no back-end, as mesmas entradas são fornecidas à criptografia de AWS pagamento, o criptograma é recriado internamente e comparado com o valor fornecido com a transação. Nesse sentido, é semelhante a um MAC. O EMV 4.4 Book 2 define três aspectos dessa função: métodos de derivação de chave (conhecidos como chave de sessão comum - CSK) para gerar chaves de transação únicas, uma carga útil mínima e métodos para gerar uma resposta (ARPC).

Os esquemas de cartões individuais podem especificar campos transacionais adicionais a serem incorporados ou a ordem em que esses campos aparecem. Outros esquemas de derivação específicos de esquemas (geralmente obsoletos) também existem e são abordados em outras partes desta documentação.

Para obter mais informações, consulte VerifyCardValidationData o guia da API.

Crie a chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/pw3s6nl62t5ushfk Isso será necessário na próxima etapa.

Gere um ARQC

O ARQC é gerado exclusivamente por um cartão EMV. Dessa forma, a criptografia de AWS pagamento não tem facilidade para gerar essa carga útil. Para fins de teste, várias bibliotecas estão disponíveis on-line que podem gerar uma carga útil apropriada, bem como valores conhecidos que geralmente são fornecidos pelos vários esquemas.

Validar um ARQC

Example

Se a criptografia AWS de pagamento conseguir validar o ARQC, um será devolvido. http/200 Opcionalmente, um ARPC (resposta) pode ser fornecido e incluído na resposta após a validação do ARQC.

```
$ 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
 00000000170000000000000008400080008000084016051700000000093800000B1F220103000000000000000000000
 --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"
}
```

Gere e verifique um MAC EMV

EMV MAC é MAC usando uma entrada de uma chave derivada de EMV e, em seguida, executando um MAC ISO9797-3 (de varejo) sobre os dados resultantes. O EMV MAC é normalmente usado para enviar comandos para um cartão EMV, como scripts de desbloqueio.

> **Note**
>
> AWS A criptografia de pagamento não valida o conteúdo do script. Consulte o esquema ou o manual do cartão para obter detalhes sobre os comandos específicos a serem incluídos.

Para obter mais informações, consulte MacAlgorithmEmv o guia da API.

Tópicos

- Crie a chave
- Gere um MAC EMV

Crie a chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
        }
    }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/pw3s6nl62t5ushfk Isso será necessário na próxima etapa.

Gere um MAC EMV

O fluxo típico é que um processo de back-end gere um script EMV (como desbloqueio de cartão), assine-o usando esse comando (que deriva uma chave única específica para uma placa específica) e, em seguida, retorne o MAC. Em seguida, o comando + MAC é enviado para a placa a ser aplicada. O envio do comando para o cartão está fora do escopo da criptografia de AWS pagamento.

> **Note**
>
> Esse comando é destinado a comandos quando nenhum dado criptografado (como PIN) é enviado. O EMV Encrypt pode ser combinado com esse comando para acrescentar dados criptografados ao script do emissor antes de chamar esse comando.

Dados da mensagem

Os dados da mensagem incluem o cabeçalho e o comando APDU. Embora isso possa variar de acordo com a implementação, este exemplo é o cabeçalho APDU para desbloqueio (84 24 00 00 08), seguido pelo ATC (0007) e depois pelo ARQC da transação anterior (999E57FD0F47CACE). O serviço não valida o conteúdo desse campo.

Modo de derivação da chave de sessão

Esse campo define como a chave da sessão é gerada. O EMV_COMMON_SESSION_KEY geralmente é usado para as novas implementações, enquanto o EMV2000 | AMEX | MASTERCARD_SESSION_KEY | VISA também pode ser usado.

MajorKeyDerivationMode

O EMV define o modo A, B ou C. O modo A é o mais comum e a criptografia AWS de pagamento atualmente suporta o modo A ou o modo B.

PAN

O número da conta, normalmente disponível no campo 5A do chip ou no campo 2 da ISO8583, mas também pode ser recuperado do sistema de cartões.

PSN

O número de sequência do cartão. Se não for usado, insira 00.

SessionKeyDerivationValue

Esses são os dados de derivação por sessão. Pode ser o último ARQC (ApplicationCryptogram) do campo 9F26 ou o último ATC do 9F36, dependendo do esquema de derivação.

Padding

O preenchimento é aplicado automaticamente e usa o método de preenchimento ISO/IEC 9797-1 2.

Example

```
$ aws payment-cryptography-data generate-mac --message-data
 842400000800079 99E57FD0F47CACE --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"
}
```

Gere EMV MAC para alteração de PIN

A alteração do PIN EMV combina duas operações: gerar um MAC para um script do emissor e criptografar um novo PIN para alteração offline do PIN em um cartão com chip EMV. Esse comando só é necessário em alguns países onde o PIN está armazenado no cartão com chip (isso é comum em países europeus). Isso geralmente é usado quando o titular do cartão precisa alterar seu PIN e o novo PIN deve ser transmitido com segurança ao cartão junto com um MAC para verificar a autenticidade do comando.

> **Note**
>
> Se você só precisar enviar comandos para a placa, mas não alterar o PIN, considere usar os comandos ARPC CSU ou Generate EMV MAC.

Para obter mais informações, consulte GenerateMacEmvPinChange o guia da API.

Gere EMV MAC e PIN criptografado para alteração de PIN

Essa operação requer duas chaves: uma chave de integridade EMV (: TR31_E2_EMV_MKEY_INTEGRITY) para geração de MAC e uma chave de confidencialidade EMV (KeyUsage: TR31_E4_EMV_MKEY_CONFIDENTIALITY) para criptografia de PIN. KeyUsage O fluxo típico é que um processo de back-end gere um script de alteração de PIN EMV, que inclui o MAC para o script do emissor e o novo PIN criptografado. O comando e o PIN criptografado são então enviados ao cartão para atualizar o PIN offline. O envio do comando para o cartão está fora do escopo da criptografia de AWS pagamento.

Dados da mensagem

Os dados da mensagem incluem o comando APDU para o script do emissor. O serviço não valida o conteúdo desse campo.

Novo bloco de PIN criptografado

O novo bloco de PIN criptografado que será enviado para o cartão. Isso deve ser fornecido como um valor criptografado usando uma chave de criptografia PIN.

Novo identificador PIN PEK

A chave usada para criptografar o novo PIN antes de ele ser passado para essa API.

Chave de integridade de mensagens seguras

A chave de integridade EMV (KeyUsage: TR31_E2_EMV_MKEY_INTEGRITY) usada para geração de MAC.

Chave de confidencialidade de mensagens seguras

A chave de confidencialidade EMV (KeyUsage: TR31_E4_EMV_MKEY_CONFIDENTIALITY) usada para criptografia de PIN.

MajorKeyDerivationMode

O EMV define o Modo A, B ou C. O modo A é o mais comum e a criptografia de AWS pagamento atualmente suporta o modo A ou o modo B.

Modo

O modo de criptografia, normalmente CBC para operações de alteração de PIN.

PAN

O número da conta, normalmente disponível no campo 5A do chip ou no campo 2 da ISO8583, mas também pode ser recuperado do sistema de cartões.

PanSequenceNumber

O número de sequência do cartão. Se não for usado, insira 00.

ApplicationCryptogram

Esses são os dados de derivação por sessão, normalmente o último ARQC do campo 9F26.

PinBlockLengthPosition

Especifica onde o comprimento do bloco PIN é codificado. Normalmente definido como NENHUM. Verifique as especificações do esquema do seu cartão se não tiver certeza.

PinBlockPaddingType

Especifica o tipo de preenchimento para o bloco de PIN. Normalmente definido como NO_PADDING. Verifique as especificações do esquema do seu cartão se não tiver certeza.

Example

```
$ aws payment-cryptography-data generate-mac-emv-pin-change \
    --message-data 00A4040008A000000004101080D80500000001010A04000000000000 \
    --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"
}
```

Funções específicas da rede

Tópicos

- Funções específicas do visto
- Funções específicas da Mastercard
- Funções específicas do American Express
- Funções específicas do JCB
- UnionPay funções específicas

Funções específicas do visto

Tópicos

- ARQC - CVN18/CVN22
- ARQC - CVN10
- 3DS CAVV V7
- dCvV (valor dinâmico de verificação de cartão) - CVN17

ARQC - CVN18/CVN22

O CVN18 e o CVN22 utilizam o método CSK de derivação de chave. Os dados exatos da transação variam entre esses dois métodos. Consulte a documentação do esquema para obter detalhes sobre a construção do campo de dados da transação.

ARQC - CVN10

O CVN10 é um método Visa mais antigo para transações EMV que usa derivação de chave por cartão em vez de derivação por sessão (por transação) e também usa uma carga útil diferente. Para obter informações sobre o conteúdo da carga útil, entre em contato com o esquema para obter detalhes.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```


Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/pw3s6nl62t5ushfk Isso será necessário na próxima etapa.

Validar o ARQC

Example

Neste exemplo, validaremos um ARQC gerado usando Visa CVN10.

Se a criptografia AWS de pagamento conseguir validar o ARQC, um será devolvido. http/200 Se o arqc não for validado, ele retornará uma resposta. 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"
  }
```

3DS CAVV V7

Para transações Visa Secure (3DS), um CAVV (Cardholder Authentication Verification Value) é gerado pelo servidor de controle de acesso (ACS) do emissor. O CAVV é uma evidência de que a autenticação do titular do cartão ocorreu, é exclusivo para cada transação de autenticação e é fornecido pelo adquirente na mensagem de autorização. O CAVV v7 vincula dados adicionais sobre a transação à aprovação, incluindo elementos como nome do comerciante, valor da compra e data da compra. Dessa forma, é efetivamente um hash criptográfico da carga útil da transação.

Criptograficamente, o CAVV V7 utiliza o algoritmo CVV, mas todas as entradas foram. changed/repurposed Consulte a terceira party/Visa documentação apropriada sobre como produzir as entradas para gerar uma carga útil CAVV V7.

Crie a chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```


Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/dnaeyrjgdjjtw6dk Isso será necessário na próxima etapa.

Gere um CAVV V7

Example

Neste exemplo, geraremos um CAVV V7 para uma determinada transação com entradas conforme especificado nas especificações. Observe que, para esse algoritmo, os campos podem ser used/re reutilizados, portanto, não se deve presumir que os rótulos dos campos correspondam às entradas.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 o guia de referência da 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"
            }
```

Validar o CAVV V7

Example

Para validação, as entradas são CVK, os valores de entrada calculados e o CAVV fornecidos durante a transação para validação.

Para ver todos os parâmetros disponíveis, CardVerificationValue1 consulte o guia de referência da API.

> **Note**
>
> O CAVV não é um valor inserido pelo usuário (como CVV2), mas é calculado pelo emissor ACS. Deve-se considerar se ele deve sempre ser validado quando fornecido.

```
$ 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 (valor dinâmico de verificação de cartão) - CVN17

O dCVV (valor dinâmico de verificação de cartão) é um criptograma Visa-specific dinâmico usado para transações EMV sem contato. É conhecido como EMV inicial e oferece segurança aprimorada ao gerar um valor de verificação exclusivo para cada transação. O DCvv usa entradas que incluem o número da conta primária (PAN), o número de sequência do PAN (PSN), o contador de transações do aplicativo (ATC), a data de validade do cartão e o código do serviço. Ele ainda é usado em alguns lugares, mas foi substituído principalmente por outros algoritmos como o CVN18.

Para ver todos os parâmetros disponíveis, consulte DynamicCardVerificationValue o guia de referência da API.

As entradas a serem `DynamicCardVerificationValue` mapeadas para os seguintes dados de autorização Visa e transação EMV:

`ApplicationTransactionCounter`(ATC)

O contador de transações mantido pelo chip, incrementado para cada transação. Nos dados EMV, isso é tag`9F36`.

`PanSequenceNumber`

O número de sequência PAN que distingue os cartões que compartilham o mesmoPAN. Nos dados EMV, isso é tag`5F34`.

`CardExpiryDate`

A data de validade do pedido do cartão (etiqueta EMV`5F24`), fornecida no mesmo formato usado por suas outras chamadas de verificação do cartão.

`ServiceCode`

O código de serviço de três dígitos contido nos dados da faixa 2 (equivalente) do cartão.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/mw7dn3qxvkfh8ztc Isso será necessário na próxima etapa.

Gere um DCvv

Example

Neste exemplo, geraremos um dCVv para uma transação EMV sem contato. As entradas incluem o PAN, o número de sequência do PAN, o contador de transações do aplicativo, a data de validade do cartão e o código do serviço.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
    --primary-account-number=4111112627662122 \
    --generation-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,CardExpir
 \
    --validation-data-length 5
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2",
    "ValidationData": "36667"
}
```

Validar DCvV

Example

Neste exemplo, validaremos um DCvv fornecido durante uma transação. As mesmas entradas usadas para geração devem ser fornecidas para validação.

Se a criptografia AWS de pagamento puder ser validada, um http/200 será devolvido. Se o valor não for validado, ele retornará uma http/400 resposta.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
    --primary-account-number=4111112627662122 \
    --validation-data=36667 \
    --verification-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,CardExpir
```

```
{
```

```
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2"
}
```

Funções específicas da Mastercard

Tópicos

- DCVC3
- ARAC -/ CVN14CVN15
- ARAC -/ CVN12CVN13
- 3DS SPA2 AAV

DCVC3

DCVC3 antecede os CVN12 esquemas EMV CSK e Mastercard e representa outra abordagem para a utilização de chaves dinâmicas. Às vezes, também é reutilizado para outros casos de uso. Nesse esquema, as entradas são dados PAN, PSN, Track1/Track2, um número imprevisível e um contador de transações (ATC).

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
        }
    }
```

Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-2:111122223333:key/hrh6qgbi3sk4y3wq. Isso será necessário na próxima etapa.

Gere um DCVC3

Example

Embora normalmente DCVC3 seja gerado por um cartão com chip, ele também pode ser gerado manualmente, como neste exemplo

```
$ 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"
```

```
  }
```

Valide o DCVC3

Example

Neste exemplo, vamos validar um DCVC3. Observe que o ATC deve ser fornecido como um número hexadecimal, por exemplo, um contador de 11 deve ser representado como 000B. O serviço espera um valor de 3 dígitos; portanto DCVC3, se você armazenou um valor de 4 (ou 5) dígitos, basta truncar os caracteres à esquerda até ter 3 dígitos (por exemplo, 15321 deve resultar em um valor de dados de validação de 321).

Se a criptografia AWS de pagamento puder ser validada, um http/200 será retornado. Se o valor não for validado, ele retornará uma resposta 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"
  }
```

ARAC -/ CVN14CVN15

CVN14 e CVN15 utilize o método EMV CSK de derivação de chaves. Os dados exatos da transação variam entre esses dois métodos. Consulte a documentação do esquema para obter detalhes sobre a construção do campo de dados da transação.

ARAC -/ CVN12CVN13

CVN12 e CVN13 são métodos mais antigos específicos da Mastercard para transações EMV que incorporam um número imprevisível na derivação por transação e também usam uma carga útil diferente. Para obter informações sobre o conteúdo da carga útil, entre em contato com o esquema.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk. Isso será necessário na próxima etapa.

Valide o ARQC

Example

Neste exemplo, validaremos um ARQC gerado usando o Mastercard. CVN12

Se a criptografia AWS de pagamento for capaz de validar o ARQC, um http/200 será retornado. Se o ARQC não for validado, ele retornará uma resposta 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
 00000000170000000000000008400080008000084016051700000000093800000B1F220103000000000000000000000
 \
  --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 SPA2 AAV

SPA2(Aplicativo de pagamento seguro) O AAV (Account Authentication Value) é usado para transações do Mastercard 3DS (também conhecido como Mastercard Idenity Check). Ele fornece autenticação criptográfica para transações de comércio eletrônico usando a geração MAC baseada em HMAC. O AAV é gerado usando dados específicos da transação e uma chave secreta compartilhada.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```


Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-west -2:111122223333:key/q5vjtshsg67cz5gn. Isso será necessário na próxima etapa.

Gerar SPA2 AAV

Example

Neste exemplo, geraremos o componente Issuer Authentication Value (IAV) do SPA2 AAV usando a geração HMAC MAC. Os dados da mensagem contêm as informações específicas da transação

que serão autenticadas. O formato dos dados da mensagem deve seguir as SPA2 especificações da Mastercard e não é abordado neste exemplo.

> **Note**
>
> Verifique as especificações da Mastercard quanto à formatação para inserir o IAV no valor do 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"
}
```

Verifique o SPA2 AAV

Example

Neste exemplo, verificaremos um SPA2 AAV. Os mesmos dados da mensagem e o mesmo valor MAC são fornecidos para verificação.

Se a criptografia AWS de pagamento conseguir validar o MAC, um http/200 será retornado. Se o MAC não for validado, ele retornará uma resposta 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"
}
```

Funções específicas do American Express

Tópicos

- CSC1
- CSC2
- iCSC
- AV 3DS

CSC1

A versão 1 do CSC também é conhecida como o algoritmo clássico do CSC. O serviço pode fornecê-lo como um número de 3,4 ou 5 dígitos.

Para ver todos os parâmetros disponíveis, consulte AmexCardSecurityCodeVersion1 no guia de referência da API.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
            }
        }
```

Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-2:111122223333:key/esh6hn7pxdtttzgq. Isso será necessário na próxima etapa.

Gere um 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"
  }
```

Valide o CSC1

Example

Neste exemplo, vamos validar um CSC1.

Se a criptografia AWS de pagamento puder ser validada, um http/200 será retornado. Se o valor não for validado, ele retornará uma resposta 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

A versão 2 do CSC também é conhecida como algoritmo CSC aprimorado. O serviço pode fornecê-lo como um número de 3,4 ou 5 dígitos. O código de serviço para normalmente CSC2 é 000.

Para todos os parâmetros disponíveis, consulte AmexCardSecurityCodeVersion2 no guia de referência da API.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
        }
    }
```

Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda. Isso será necessário na próxima etapa.

Gere um CSC2

Neste exemplo, geraremos um CSC2 com um comprimento de 4. O CSC pode ser gerado com um comprimento de 3,4 ou 5. Para American Express, PANs deve ter 15 dígitos e começar com 34 ou 37. A data de expiração geralmente é formatada como YYMM. O código de serviço pode variar - revise seu manual, mas os valores típicos são 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"
}
```

Valide o CSC2

Example

Neste exemplo, vamos validar um CSC2.

Se a criptografia AWS de pagamento puder ser validada, um http/200 será retornado. Se o valor não for validado, ele retornará uma resposta 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

O iCSC também é conhecido como algoritmo CSC estático e é calculado usando o CSC versão 2. O serviço pode fornecê-lo como um número de 3,4 ou 5 dígitos.

Use o código de serviço 999 para calcular o iCSC para um cartão de contato. Use o código de serviço 702 para calcular o iCSC para um cartão sem contato.

Para todos os parâmetros disponíveis, consulte AmexCardSecurityCodeVersion2 no guia de referência da API.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
        }
    }
```


Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv. Isso será necessário na próxima etapa.

Gere um iCSC

Neste exemplo, geraremos um iCSC com um comprimento de 4, para um cartão sem contato usando o código de serviço 702. O CSC pode ser gerado com um comprimento de 3,4 ou 5. Para American Express, PANs deve ter 15 dígitos e começar com 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"
}
```

Validar o iCSC

Example

Neste exemplo, validaremos um iCSC.

Se a criptografia AWS de pagamento puder ser validada, um http/200 será retornado. Se o valor não for validado, ele retornará uma resposta 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
}
```

AV 3DS

O 3DS AEVV (3-D Secure Account Verification Value) é usado para a autenticação American Express 3-D Secure. Ele usa o mesmo algoritmo, CSC2 mas com parâmetros de entrada diferentes. O campo da data de expiração deve ser preenchido com um número imprevisível (aleatório), e o

código do serviço consiste no Código de Resultados da Autenticação AEVV (1 dígito) mais o Código de Autenticação de Segundo Fator (2 dígitos). O comprimento da saída deve ser de 3 dígitos.

Para todos os parâmetros disponíveis, consulte AmexCardSecurityCodeVersion2 no guia de referência da API.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
        }
    }
```

Anote o `KeyArn` que representa a chave, por exemplo, arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm. Isso será necessário na próxima etapa.

Gere um 3DS AEVV

Neste exemplo, geraremos um 3DS AEVV com um comprimento de 3. O campo da data de expiração contém um número imprevisível (aleatório) (por exemplo, 1234), e o código do serviço consiste no Código de Resultados de Autenticação AEVV (1 dígito) mais o Código de Autenticação de Segundo Fator (2 dígitos), por exemplo, 543, onde 5 é o Código de Resultados de Autenticação e 43 é o Código de Autenticação de Segundo Fator. Para American Express, PANs deve ter 15 dígitos e começar com 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"
}
```

Valide o 3DS AEVV

Example

Neste exemplo, validaremos um 3DS AEVV.

Se a criptografia AWS de pagamento puder ser validada, um http/200 será retornado. Se o valor não for validado, ele retornará uma resposta 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
}
```

Funções específicas do JCB

Tópicos

- ARQC - CVN04
- ARQC - CVN01

ARQC - CVN04

O JCB CVN04 utiliza o método CSK de derivação de chave. ??? Consulte a documentação do esquema para obter detalhes sobre a construção do campo de dados da transação.

ARQC - CVN01

O CVN01 é um método JCB mais antigo para transações EMV que usa derivação de chave por cartão em vez de derivação por sessão (por transação) e também usa uma carga útil diferente. Essa mensagem também é usada pela Visa, portanto, o nome do elemento tem esse nome, embora também seja usado para JCB. Para obter informações sobre o conteúdo da carga útil, entre em contato com a documentação do esquema.

Criar chave

```
$ 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"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes, bem como um valor de verificação chave (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"
                    }
                }
```

Anote o `KeyArn` que representa a chave, por exemplo arn:aws:payment-cryptography:us-east-2:111122223333:. key/pw3s6nl62t5ushfk Isso será necessário na próxima etapa.

Validar o ARQC

Example

Neste exemplo, validaremos um ARQC gerado usando o JCB CVN01. Isso usa as mesmas opções do método Visa, daí o nome do parâmetro.

Se a criptografia AWS de pagamento conseguir validar o ARQC, um será devolvido. http/200 Se o arqc não for validado, ele retornará uma resposta. 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"
              }
```

UnionPay funções específicas

Os exemplos a seguir referem-se à UnionPay International (UPI), também conhecida como China UnionPay (CUP).

Tópicos

- Validação ARQC
- CVN2 (Número de verificação do cartão)
- AVN (número de verificação de autenticação)

Validação ARQC

UnionPay A validação do ARQC usa o modo de derivação de chave EMV_OPTION_A. A chave da sessão é derivada do PrimaryAccountNumber Número de Sequência PAN e do Contador de Transações do Aplicativo usando um UnionPay-specific esquema. Para obter informações sobre o formato dos dados da transação e a ordem dos campos, consulte a documentação do esquema.

Validar o ARQC

Example Validar um ARQC usando derivação de chave UnionPay

Este exemplo mostra como validar um ARQC gerado usando UnionPay derivação de chave.

Se a criptografia AWS de pagamento validar o ARQC, ela retornará uma resposta HTTP 200. Se o ARQC falhar na validação, a criptografia AWS de pagamento retornará uma resposta HTTP 400 com um código de motivo.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram A12345B1A12345B1 \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
  --major-key-derivation-mode EMV_OPTION_A \
  --transaction-data 123456789ABCDEF80000000000000000 \
  --session-key-derivation-
attributes='UnionPay={PrimaryAccountNumber=4123412341234123,PanSequenceNumber=01,ApplicationTra
```

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

Se o ARQC não puder ser validado, a criptografia de AWS pagamento retornará uma resposta de erro:

```
An error occurred (VerificationFailedException) when calling the
 VerifyAuthRequestCryptogram operation: Auth request cryptogram verification failed.

Reason: INVALID_AUTH_REQUEST_CRYPTOGRAM
```

CVN2 (Número de verificação do cartão)

CVN2 é UnionPay equivalente ao CVV2. É um número de verificação de cartão usado para transações on-line e sem cartão presente. Criptograficamente, o CVN2 usa o mesmo algoritmo do CVV2 (). CardVerificationValue2 As entradas são o Número da Conta Primária (PAN) e a data de validade do cartão (4 dígitos no formato MMYY).

Crie a chave

Para criar uma chave CVN2, use o `create-key` comando com o uso e o algoritmo`TR31_C0_CARD_VERIFICATION_KEY`. `TDES_2KEY`

```
$ aws payment-cryptography create-key --exportable \
  --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModes0
 \
  --tags='[{"Key":"KEY_PURPOSE","Value":"CVN2"},{"Key":"CARD_BIN","Value":"47613499"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes e um Key Check Value (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"
    }
}
```


Anote o `KeyArn` valor, por exemplo arn:aws:payment-cryptography:us-east- 2:111122223333:. key/7f7g4spf3xcklhzu Você precisa disso nas etapas a seguir.

Gere um CVN2

Example Gere um CVN2 para um determinado PAN

Este exemplo gera um CVN2 para um determinado PAN com entradas `PAN` e data de validade do cartão.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue2 o guia de referência da API.

```
$ aws payment-cryptography-data generate-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu \
  --primary-account-number=4761349999991234 \
  --generation-attributes CardVerificationValue2='{CardExpiryDate=1226}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
    "KeyCheckValue": "AEA5CD",
    "ValidationData": "321"
}
```

Validar um CVN2

Example Validar um CVN2 para um determinado PAN

Este exemplo valida um CVN2 para um determinado PAN com entradas dePAN, data de validade do cartão e o CVN2 fornecido durante a transação.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue2 o guia de referência da API.

> **Note**
>
> CVN2 é um valor inserido pelo usuário. Falhas periódicas de validação não indicam necessariamente um problema.

```
$ aws payment-cryptography-data verify-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu \
  --primary-account-number=4761349999991234 \
  --verification-attributes CardVerificationValue2='{CardExpiryDate=1226}' \
  --validation-data 321
```

```
{
```

```
    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
    "KeyCheckValue": "AEA5CD"
}
```

AVN (número de verificação de autenticação)

AVN (Número de Verificação de Autenticação) é o criptograma UnionPay 3DS. Ele usa o algoritmo CVV (CardVerificationValue1) com campos de entrada reaproveitados, como abaixo, mas verifique a documentação do esquema para obter detalhes finalizados.

- PAN — Número da conta primária (16-19 dígitos, da Área Q)
- CardExpiryDate— Os quatro dígitos menos significativos de um valor derivado da ID da transação DS (dsTransid)
- ServiceCode— 3 dígitos: Resultado da transação de autenticação (1 dígito) + bit de preenchimento '0' (1 dígito) +indicador do gerador AV (1 dígito)

Crie a chave

Para criar uma chave AVN, use o `create-key` comando com o `TR31_C0_CARD_VERIFICATION_KEY` uso e o `TDES_2KEY` algoritmo.

```
$ 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":"AVN"},{"Key":"CARD_BIN","Value":"47613499"}]'
```

A resposta reflete os parâmetros da solicitação, incluindo um ARN para chamadas subsequentes e um Key Check Value (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"
    }
}
```

Anote o `KeyArn` valor, por exemplo arn:aws:payment-cryptography:us-east- 2:111122223333:. key/ dnaeyrjgdjjtw6dk Você precisa disso nas etapas a seguir.

Gere um AVN

Example Gere um AVN para um determinado PAN

Este exemplo gera um AVN para um determinado PAN. O `CardExpiryDate` campo contém os quatro dígitos menos significativos derivados do dsTransid. O `ServiceCode` campo contém o resultado da autenticação, o bit de preenchimento e o indicador do gerador.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 o guia de referência da API.

```
$ aws payment-cryptography-data generate-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk \
  --primary-account-number=4761349999991234 \
  --generation-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=201}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
    "KeyCheckValue": "F3FB13",
    "ValidationData": "491"
}
```

Validar um AVN

Example Validar um AVN para um determinado PAN

Este exemplo valida um AVN para um determinado PAN usando os mesmos campos de entrada que o geraram, junto com o valor do AVN a ser verificado.

Para ver todos os parâmetros disponíveis, consulte CardVerificationValue1 o guia de referência da API.

```
$ aws payment-cryptography-data verify-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk \
  --primary-account-number=4761349999991234 \
  --verification-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=201}' \
  --validation-data 491
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
    "KeyCheckValue": "F3FB13"
}
```

Facilitadores de aquisição e pagamento

Adquirentes PSPs e facilitadores de pagamento normalmente têm um conjunto de requisitos criptográficos diferente dos emissores. Entre os casos de uso comuns estão:

Descriptografia de dados

Os dados (especialmente os dados bancários) podem ser criptografados por um terminal de pagamento e precisam ser descriptografados pelo back-end. O Decrypt Data e o Encrypt Data

suportam uma variedade de métodos, incluindo técnicas de derivação de TDES, AES e DUKPT. O serviço AWS de criptografia de pagamento em si também é compatível com PCI P2PE e está registrado como um componente de decodificação PCI P2PE.

TranslatePin

Para manter a conformidade com o PIN PCI, os sistemas adquirentes não devem ter os pinos do titular do cartão transparentes após terem sido inseridos em um dispositivo seguro. Portanto, para passar o PIN do terminal para um sistema posterior (como uma rede de pagamento ou emissor), é necessário criptografá-lo novamente usando uma chave diferente da usada pelo terminal de pagamento. O Translate Pin faz isso convertendo um PIN criptografado de uma chave para outra de forma segura com o servicebbb. Usando esse comando, os pinos podem ser convertidos entre vários esquemas, como derivação TDES, AES e DUKPT, e formatos de blocos de pinos, como ISO-0, ISO-3 e ISO-4.

VerifyMac

Os dados de um terminal de pagamento podem ser maculados para garantir que os dados não tenham sido modificados em trânsito. Verifique o Mac e GenerateMac oferece suporte a uma variedade de técnicas usando chaves simétricas, incluindo técnicas de derivação TDES, AES e DUKPT para uso com as técnicas ISO-9797-1, ISO-9797-1 (MAC de varejo) e CMAC.

Tópicos adicionais

- Usando teclas dinâmicas

Usando teclas dinâmicas

As chaves dinâmicas permitem que chaves de uso único ou limitado sejam usadas para operações criptográficas, como. `EncryptData` Esse fluxo pode ser utilizado quando o material-chave gira com frequência (como em todas as transações com cartão) e há o desejo de evitar a importação do material-chave para o serviço. Chaves de curta duração podem ser utilizadas como parte do SoftPOS/MPOC ou de outras soluções.

> **Note**
>
> Isso pode ser usado no lugar do fluxo típico usando criptografia de AWS pagamento, em que as chaves criptográficas são criadas ou importadas para o serviço e as chaves são especificadas usando um alias de chave ou arn de chave.

As operações a seguir oferecem suporte a teclas dinâmicas:

- EncryptData
- DecryptData
- ReEncryptData
- TranslatePin

Descriptografia de dados

O exemplo a seguir mostra o uso de chaves dinâmicas junto com o comando decrypt. O identificador de chave nesse caso é a chave de encapsulamento (KEK) que protege a chave de decodificação (que é fornecida no parâmetro de chave encapsulada no formato TR-31). A chave encapsulada deve ser o objetivo principal de D0 a ser usada com o comando decrypt junto com um modo de uso 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"
}
```

Traduzindo um alfinete

O exemplo a seguir mostra o uso de teclas dinâmicas junto com o comando translate pin para traduzir de uma chave dinâmica para uma chave de trabalho semiestática do adquirente (AWK). O identificador de chave de entrada nesse caso é a chave de encapsulamento (KEK) que protege a chave de criptografia dinâmica de pinos (PEK) fornecida no formato TR-31. A chave encapsulada deve ser o objetivo principal de, P0 juntamente com um modo de uso de B ou D. O identificador

de chave de saída é uma chave do tipo TR31_P0_PIN_ENCRYPTION_KEY e um modo de uso 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"
}
```

Características específicas da região para AWS Payment Cryptography

Certos recursos podem ser específicos da região e não serem usados de outra forma. Esses recursos são descritos com mais detalhes nesta seção.

AS2805

O Australia Standard 2805 (AS2805) é um padrão para transferências eletrônicas de fundos usado principalmente para transações de pagamento baseadas em cartões. É mantido pela Standards Australia. O padrão consiste em 6 livros que abordam vários tópicos, desde formato de mensagem até padrões de criptografia.

A Parte 6 fornece orientação sobre gerenciamento de chaves, incluindo comunicação de host a host (nó a nó) e requisitos criptográficos relevantes, enquanto outros aspectos são abordados em outras partes. Atualmente, toda a criptografia nesse padrão é baseada no TDES.

O AS2805 tem várias diferenças em comparação com outras implementações, que estão resumidas abaixo.

Proteção de chaves

Depende de variantes de teclas em vez de blocos de teclas, como in. TR-31/X9.143 AWS A criptografia de pagamento armazena todas as chaves como blocos de chaves internamente, mas permite importação, exportação e cálculo usando as variantes definidas pelo AS2805.

Teclas unidirecionais

O AS2805 exige o uso de chaves unidirecionais. Se os dois nós precisarem gerar códigos de autenticação de mensagens (MAC), eles usarão duas chaves.

Blocos de pinos

O AS2805 define uma técnica de derivação de chave para chaves de criptografia de pinos exclusivas por transação. Isso pode ser usado no lugar do DUKPT. O esquema AS2805 se baseia nos dados da transação (número de rastreamento e valor da transação) em comparação com o uso do contador de transações pela DUKPT.

Validação de troca de chaves

Define um processo para validar o KEK antes de começar a trocar chaves de trabalho, como chaves PIN. Em outros esquemas, as KEK são trocadas com pouca frequência e são validadas usando o KCV.

O AS2805 usa o conceito de variantes de chave em vez de blocos de chaves para garantir que as chaves sejam usadas apenas para a finalidade pretendida (e única). Veja a seguir como a criptografia AWS de pagamento mapeia entre variantes e blocos de teclas ao importar, exportar ou executar outras funções criptográficas com chaves.

Tipo de chave AS2805	AWS Tipo de chave de criptografia de pagamento
TERMINAL_MAJOR_KEY_VARIANT_00	TR31_K0_KEY_ENCRYPTION_KEY
PIN_ENCRYPTION_KEY_VARIANT_28	TR31_P0_PIN_ENCRYPTION_KEY
MENSAGEM_AUTENTICAÇÃO_CHAVE _VARIANT_24	TR31_M0_ISO_16609_MAC_KEY
VARIANTE DA CHAVE DE CRIPTOGRAFIA DE DADOS_22	TR31_D0_SYMMETRIC_DATA_ENCR YPTION_KEY
VARIANTE_MASK_82, VARIANTE_ MASK_82C0	Opções disponíveis como parte do processo de validação do KEK. Esses tipos de chave são efêmeros e não são armazenados pelo serviço.

Dados dois nós, node1 e node2, os exemplos a seguir são da perspectiva de node1. AWS A criptografia de pagamento suporta APIs de ambos os lados do processo.

Tópicos

- Troca de chave inicial (KEK)
- Validação do KEK
- Criação e transmissão de chaves de trabalho
- Exportação de chaves de trabalho
- Tradução de Pin

- Geração e validação de Mac

Troca de chave inicial (KEK)

No AS2805, cada lado tem seu próprio KEK. KEK (s) se refere à chave do lado de envio que será usada sempre que o lado remetente precisar de protect/wrap chaves e enviá-las para o node2. KEK (r) é a chave criada pelo lado oposto (node2).

> **Note**
>
> Esses termos são relativos - um lado cria uma chave (lado remetente) e o outro a recebe. Portanto, dada KEY1, ela é referida no node1 como KEK (s) e no node2 como KEK (r).

As KEK para AS2805 são sempre do tipo chave = TR31_K0_KEY_ENCRYPTION_KEY, pois são usadas para proteger criptogramas e não blocos de chaves. Isso mapeia para TERMINAL_MAJOR_KEY_VARIANT_00 conforme definido no AS2805 6.1

Etapas:

1. Crie uma chave

   Crie uma chave usando a CreateKey API. Você criará uma chave do tipo TR31_K0_KEY_ENCRYPTION_KEY

2. Determine o método para trocar chaves com o node2

   Determine como trocar KEK com a contraparte. Para o AS2805, o método mais comum e interoperável é o RSA Wrap.

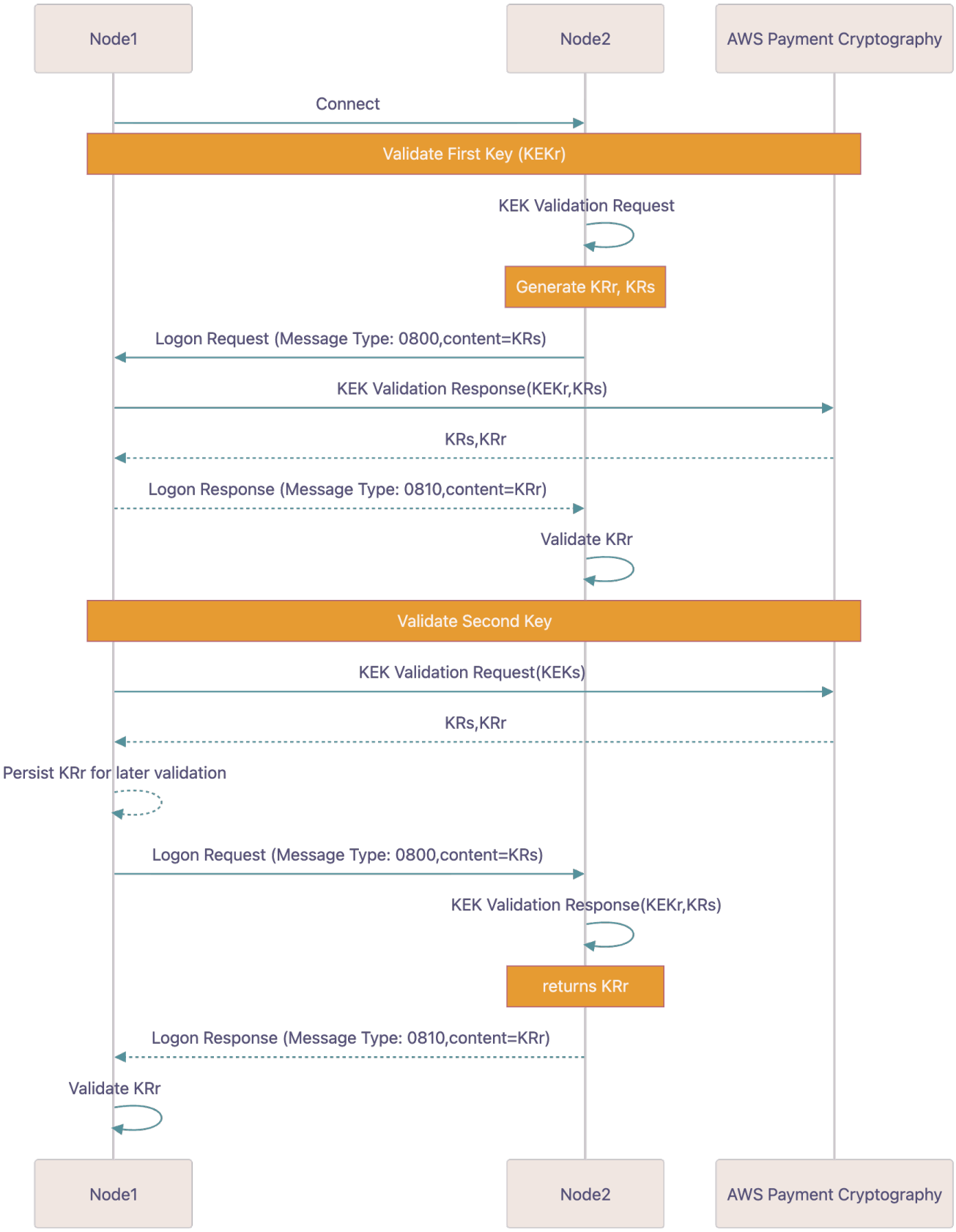
3. Exportar KEKs

   Com base na sua seleção acima, você receberá um certificado de chave pública do node2. Você executará a exportação usando esse certificado para proteger a chave (ou derivará uma chave se estiver usando ECDH).

4. Importar KEKr

   Com base na sua seleção acima, você enviará um certificado de chave pública para o node2. Você executará a importação usando esse certificado para carregar o KEkr do nó 2 no serviço.

Validação do KEK



Quando seu serviço (node1) se conecta ao node2, cada lado garante que eles estejam usando a mesma KEK para operações subsequentes usando um processo chamado Validação KEK.

1. Etapas para validar a primeira chave

1.1 Receba KRs

O Node2 gerará um KRs e o enviará para você como parte do processo de login. Eles podem usar a criptografia de AWS pagamento para gerar esse valor ou outra solução.

1.2 Gerar resposta de validação KEK

Seu nó gerará uma resposta de validação de KEK com entradas como KEK (r) e KRs fornecidas na etapa 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 Retorno calculado em KrR

Retorne o kRR calculado para node2. Esse nó o comparará com o valor calculado da etapa 1.

2.Etapas para validar a segunda chave

2.1 Gere KRr e KRs

Seu node gerará um valor aleatório e uma cópia invertida (invertida) desse valor usando a criptografia de AWS pagamento. O serviço produzirá esses dois valores agrupados pelo (s) KEK (s). Eles são conhecidos como KR (s) e 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 Envie KRs para o node2

Envie os KRs para o node2. Guarde o KrR para posterior validação.

2.3 O Node2 gera resposta de validação KEK

O Node2 usa o KeKr e o KRs, gera o KrR e o envia de volta ao seu serviço.

2.4 Validar a resposta

Compare o KrR da etapa 1 e o valor retornado da etapa 3. Se eles coincidirem, prossiga.

Criação e transmissão de chaves de trabalho

As chaves de trabalho típicas usadas no AS2805 incluem dois conjuntos de chaves:

Chaves entre nós, como: chave de pino de zona (ZPK), chave de criptografia de zona (ZEK) e chave de autenticação de zona (ZAK).

Teclas entre terminais e nós, como: chave principal do terminal (TMK) e chave do pino do terminal (TPK) se não estiver usando o DUKPT.

> **Note**
>
> Recomendamos minimizar chaves por chave de terminal e aproveitar técnicas como TR-34 DUKPT sempre que possível, que usam um número menor de chaves.

Example

Neste exemplo, usamos tags opcionais para rastrear a finalidade e o uso dessa chave. As tags não são usadas como parte da função criptográfica do sistema, mas podem ser usadas para categorização, rastreamento financeiro e podem ser usadas para aplicar políticas do 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 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"
 }
}
```

Exportação de chaves de trabalho

Para manter a compatibilidade com outras partes, a criptografia de AWS pagamento oferece suporte às técnicas de empacotamento simétrico de chaves AS2805 que usam variantes de chave em vez de blocos de teclas, como. TR-31 Se várias chaves forem compartilhadas entre as partes, cada uma deverá ser exportada individualmente. Se os dados forem enviados bidirecionalmente, pode haver duas chaves entre partes do mesmo tipo, como ZAK (s) e ZAK (r), que são usadas por cada lado para gerar códigos de autenticação de mensagens.

Os parâmetros adicionais para importar e exportar nesses formatos são especificados nos comandos.

```
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 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"
    }
}
```

Tradução de Pin

O AS2805 descreve um modo de derivação de chave específico da sessão na seção 6.4. Ele tem um propósito semelhante ao DUKPT e qualquer um dos algoritmos pode ser usado, pois o DUKPT é abordado na seção 6.7. Nesse esquema, uma chave de pino de sessão (conhecida como KPE) é derivada da chave de pinos do terminal usando SystemTraceAuditNumber (STAN) e TransactionAmount como dados de derivação.

O pino de tradução é uma função comum que pode traduzir to/from uma variedade de formatos. Neste exemplo, traduzimos um PIN de um KPE para uma chave de criptografia de pinos (PEK), como ao enviar um PIN para uma rede de pagamento.

```
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"
    }
}
```

Geração e validação de Mac

Os comandos de geração e verificação de MAC suportam uma variedade de MACs, incluindo HMAC, CMAC, EMV MAC, etc. Para o AS2805, há uma variação adicional definida em. AS2805.4.1 Normalmente, no AS2805, as mensagens recebidas são verificadas usando esse MAC e as mensagens de saída também incluem um 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"
}
```

Segurança em AWS Payment Cryptography

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como a segurança da nuvem e a segurança na nuvem:

- Segurança da nuvem —AWS é responsável por proteger a infraestrutura que executa AWS serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Third-party auditores testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS Para saber mais sobre os programas de conformidade que se aplicam à criptografia AWS de pagamentos, consulte AWS Services in Scope by Compliance Program .
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

Este tópico ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar a criptografia AWS de pagamento. Ele mostra como configurar a criptografia AWS de pagamento para atender aos seus objetivos de segurança e conformidade. Você também aprenderá a usar outros AWS serviços que ajudam você a monitorar e proteger seus recursos AWS de criptografia de pagamento.

Tópicos

- Proteção de dados em AWS Payment Cryptography
- Resiliência em AWS Payment Cryptography
- Segurança de infraestrutura em AWS Payment Cryptography
- Conectando-se a AWS Criptografia de pagamento por meio de um endpoint VPC
- Usar TLS pós-quântico híbrido
- Práticas recomendadas de segurança para AWS Payment Cryptography

Proteção de dados em AWS Payment Cryptography

O modelo de responsabilidade AWS https://aws.amazon.com/compliance/shared-responsibility-model/ se aplica à proteção de dados na criptografia AWS de pagamentos. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que opera todos os Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre privacidade de dados, consulte Perguntas frequentes sobre privacidade de dados e termos AWS da região da China. Para obter informações sobre proteção de dados na Europa, consulte o Centro de Regulamento Geral sobre a Proteção de Dados (RGPD).

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com Centro de Identidade do AWS IAM ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com AWS recursos. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre como usar CloudTrail trilhas para capturar AWS atividades, consulte Como trabalhar com CloudTrail trilhas no Guia AWS CloudTrail do usuário.
- Use soluções AWS de criptografia, junto com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sensíveis armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para saber mais sobre os endpoints FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-3.

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sensíveis, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com criptografia AWS de pagamento ou outra Serviços da AWS usando o console, a API ou os AWS SDKs. AWS CLI Quaisquer dados inseridos em tags ou em

campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

A AWS Payment Cryptography armazena e protege suas chaves de criptografia de pagamento para torná-las altamente disponíveis e, ao mesmo tempo, fornecer controle de acesso forte e flexível.

Tópicos

- Proteger material de chave
- Criptografia de dados
- Criptografia em repouso
- Criptografia em trânsito
- Privacidade do tráfego entre redes

Proteger material de chave

Por padrão, a AWS Payment Cryptography protege o material de chave criptográfica das chaves de pagamento gerenciadas pelo serviço. Além disso, a AWS Payment Cryptography oferece opções para importar material de chave criado fora do serviço. Para obter detalhes técnicos sobre chaves de pagamento e material de chave, consulte Detalhes criptográficos da AWS Payment Cryptography.

Criptografia de dados

Os dados na AWS Payment Cryptography consistem em chaves de AWS Payment Cryptography, o material de chave de criptografia que elas representam e seus atributos de uso. O material de chaves existe em texto simples apenas nos módulos de segurança de hardware (HSMs) da AWS Payment Cryptography, e somente quando em uso. Caso contrário, o material e os atributos de chave serão criptografados e armazenados de maneira persistente durável.

O material de chave que a AWS Payment Cryptography gera ou carrega para chaves de pagamento nunca deixam os limites dos HSMs da AWS Payment Cryptography sem criptografia. Ele pode ser exportado criptografado pelas operações da API de AWS Payment Cryptography.

Criptografia em repouso

O AWS Payment Cryptography gera material de chaves para chaves de pagamento em HSMs PCI PTS HSM-listed . Quando não estiver em uso, o material de chave é criptografado por uma chave do

HSM e gravado em um armazenamento durável e persistente. O material de chave para chaves de Payment Cryptography e as chaves de criptografia que protegem o material de chave nunca saem dos HSMs em formato de texto simples.

A criptografia e o gerenciamento do material de chave para chaves de criptografia de pagamento são administrados inteiramente pelo serviço.

Para obter mais detalhes, consulte Detalhes criptográficos do serviço de gerenciamento de chaves da AWS.

Criptografia em trânsito

O material de chave que a criptografia de AWS pagamento gera ou carrega para chaves de pagamento nunca é exportado ou transmitido nas operações da API de criptografia AWS de pagamento em texto não criptografado. AWS A criptografia de pagamento usa identificadores de chave para representar as chaves nas operações da API.

No entanto, algumas operações de API exportam chaves criptografadas por uma chave de troca de chaves anteriormente compartilhada ou assimétrica. Além disso, os clientes podem usar operações de API para importar material de chave criptografada para chaves de pagamento.

Todas as chamadas da API de criptografia de AWS pagamento devem ser assinadas e transmitidas usando o Transport Layer Security (TLS). AWS A criptografia de pagamento exige versões TLS e conjuntos de cifras definidos pelo PCI como “criptografia forte”. Todos os endpoints de serviço oferecem suporte ao TLS 1.2—1.3 e ao TLS híbrido pós-quântico.

Para obter mais detalhes, consulte Detalhes criptográficos do serviço de gerenciamento de chaves da AWS.

Privacidade do tráfego entre redes

AWS A criptografia de pagamento oferece suporte a um AWS Management Console e a um conjunto de operações de API que permitem criar e gerenciar chaves de pagamento e usá-las em operações criptográficas.

AWS A criptografia de pagamento oferece suporte a duas opções de conectividade de rede da sua rede privada com a AWS.

- Uma conexão VPN IPSec pela Internet.

- O AWS Direct Connect, que conecta sua rede interna a um local do AWS Direct Connect por meio de um cabo de fibra óptica Ethernet padrão.

Todas as chamadas da API de Payment Cryptography devem ser assinadas e transmitidas usando Transport Layer Security (TLS). As chamadas também exigem um conjunto de codificação moderno que seja compatível com o sigilo de encaminhamento perfeito. O tráfego para os módulos de segurança de hardware (HSMs) que armazenam material de chave para chaves de pagamento é permitido somente a partir de hosts conhecidos da API de AWS Payment Cryptography na rede interna da AWS.

Para se conectar diretamente à AWS Payment Cryptography a partir da sua nuvem privada virtual (VPC) sem enviar tráfego pela Internet pública, use os endpoints da VPC, com tecnologia da AWS. PrivateLink Para obter mais informações, consulte a Conexão à AWS Payment Cryptography por meio de um endpoint da VPC.

O AWS Payment Cryptography também oferece suporte a uma opção híbrida de troca de chaves pós-quânticas para o protocolo de criptografia de rede Transport Layer Security (TLS). É possível usar essa opção com TLS ao se conectar aos endpoints da API de AWS Payment Cryptography.

Resiliência em AWS Payment Cryptography

AWS a infraestrutura global é construída em torno de AWS regiões e zonas de disponibilidade. As regiões fornecem várias zonas de disponibilidade separadas e isoladas fisicamente, que são conectadas com baixa latência, alta throughput e redes altamente redundantes. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte Infraestrutura AWS global.

Isolamento regional

O AWS Payment Cryptography é um serviço regional que está disponível em várias regiões.

O design isolado regionalmente da AWS Payment Cryptography garante que um problema de disponibilidade em uma região da AWS não afete a operação de AWS Payment Cryptography em nenhuma outra região. A AWS Payment Cryptography foi projetada para garantir zero tempo

de inatividade planejado, com todas as atualizações de software e operações de escalabilidade realizadas de forma perfeita e imperceptível.

O Acordo de Serviço (SLA) de AWS Payment Cryptography inclui um compromisso de serviço de 99,99% para todas as APIs de Payment Cryptography. Para cumprir esse compromisso, a AWS Payment Cryptography garante que todos os dados e informações de autorização necessários para executar uma solicitação de API estejam disponíveis em todos os hosts regionais que recebem a solicitação.

A infraestrutura da AWS Payment Cryptography é replicada em pelo menos três zonas de disponibilidade (AZs) em cada região. Para garantir que várias falhas de host não afetem o desempenho da AWS Payment Cryptography, a AWS Payment Cryptography foi projetada para atender o tráfego de clientes de qualquer uma das AZs em uma região.

As alterações feitas nas propriedades ou permissões de uma chave de pagamento são replicadas para todos os hosts na região a fim de garantir que a solicitação subsequente possa ser processada corretamente por qualquer host na região. As solicitações de operações criptográficas usando sua chave de pagamento são encaminhadas para uma frota de módulos de segurança de hardware (HSMs) da AWS Payment Cryptography, e qualquer um deles pode executar a operação com a chave de pagamento.

Multi-tenant desenhar

O design de vários locatários da AWS Payment Cryptography permite cumprir o SLA de disponibilidade e sustentar altas taxas de solicitação, ao mesmo tempo que protege a confidencialidade de suas chaves e dados.

Há vários mecanismos de imposição de integridade implantados para garantir que a chave de pagamento especificada para a operação criptográfica sempre seja a chave usada.

O material de chave em texto simples para suas chaves de Payment Cryptography é amplamente protegido. Assim que é criado, o material de chave é criptografado no HSM e o material de chave criptografado é imediatamente movido para armazenamento seguro. A chave criptografada é recuperada e descriptografada no HSM no momento do uso. A chave em texto não criptografado permanece na memória do HSM apenas pelo tempo necessário para a conclusão da operação criptográfica. O material de chave em texto não criptografado nunca deixa os HSMs e nunca é gravado no armazenamento persistente.

Para obter mais informações sobre os mecanismos que a AWS Payment Cryptography usa para proteger suas chaves, consulte Detalhes criptográficos da AWS Payment Cryptography.

Segurança de infraestrutura em AWS Payment Cryptography

Como um serviço gerenciado, AWS Payment Cryptography é protegido pelos procedimentos AWS globais de segurança de rede descritos no whitepaper Amazon Web Services: Visão geral dos processos de segurança.

Você usa chamadas de API AWS publicadas para acessar AWS Payment Cryptography pela rede. Os clientes devem oferecer suporte a Transport Layer Security (TLS) 1.2 ou posterior. Os clientes também devem oferecer suporte a pacotes de cifras com sigilo direto perfeito (PFS), como Ephemeral (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Diffie-Hellman A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Isolamento de hosts físicos

A segurança da infraestrutura física usada pela AWS Payment Cryptography está sujeita aos controles descritos na seção Segurança física e ambiental da Amazon Web Services: visão geral dos processos de segurança. É possível encontrar mais detalhes em relatórios de conformidade e em descobertas de auditoria de terceiros listados na seção anterior.

O AWS Payment Cryptography é suportado por módulos de segurança de hardware (HSMs) PCI PTS HSM-listed comerciais dedicados e prontos para uso. O material de chave para chaves de AWS Payment Cryptography é armazenado somente na memória volátil dos HSMs e enquanto a chave de criptografia de pagamento estiver em uso. Os HSMs ficam em racks de acesso controlado nos datacenters da Amazon que impõem controle duplo para qualquer acesso físico. Para obter informações detalhadas sobre a operação de HSMs da AWS Payment Cryptography, consulte Detalhes criptográficos da AWS Payment Cryptography.

Conectando-se a AWS Criptografia de pagamento por meio de um endpoint VPC

Você pode se conectar diretamente à criptografia AWS de pagamento por meio de um endpoint de interface privada em sua nuvem privada virtual (VPC). Quando você usa uma interface de

VPC endpoint, a comunicação entre sua VPC e a criptografia de AWS pagamento é conduzida inteiramente dentro da rede. AWS

AWS A criptografia de pagamento é compatível com endpoints do Amazon Virtual Private Cloud (Amazon VPC) desenvolvidos por. AWS PrivateLink Cada endpoint da VPC é representado por uma ou mais interfaces de rede elástica (ENIs) com endereços IP privados em sua sub-redes da VPC.

A interface VPC endpoint conecta sua VPC diretamente à criptografia de AWS pagamento sem um gateway de internet, dispositivo NAT, conexão VPN ou conexão. AWS Direct Connect As instâncias em sua VPC não precisam de endereços IP públicos para se comunicar com a criptografia AWS de pagamento.

Regiões

AWS A criptografia de pagamento é compatível com endpoints de VPC e políticas de endpoints de VPC em todas Regiões da AWS as áreas em que AWS a criptografia de pagamento é compatível.

Tópicos

- Considerações para AWS Endpoints VPC de criptografia de pagamento
- Criação de um endpoint VPC para AWS Payment Cryptography
- Conectando-se a um AWS Endpoint VPC de criptografia de pagamento
- Controlar o acesso a um endpoint da VPC
- Usar um endpoint da VPC em uma declaração de política
- Registrar o endpoint da VPC em log

Considerações para AWS Endpoints VPC de criptografia de pagamento

> **Note**
>
> Embora os endpoints de VPC permitam que você se conecte ao serviço em apenas uma zona de disponibilidade (AZ), recomendamos conectar-se a três zonas de disponibilidade para fins de alta disponibilidade e redundância.

Antes de configurar um endpoint VPC de interface para criptografia de AWS pagamento, revise o tópico Propriedades e limitações do endpoint de interface no Guia. AWS PrivateLink

AWS O suporte à criptografia de pagamento para um endpoint VPC inclui o seguinte.

- Você pode usar seu endpoint de VPC para chamar todas as operações do plano de controle AWS de criptografia de pagamento e operações do plano de dados de criptografia AWS de pagamento a partir de uma VPC.
- Você pode criar um endpoint VPC de interface que se conecta a um endpoint da região AWS de criptografia de pagamento.
- AWS A criptografia de pagamento consiste em um plano de controle e um plano de dados. Você pode optar por configurar um ou ambos os subserviços AWS PrivateLink , mas cada um é configurado separadamente.
- Você pode usar AWS CloudTrail registros para auditar o uso das chaves de criptografia de AWS pagamento por meio do endpoint VPC. Para obter detalhes, consulte Registrar o endpoint da VPC em log.

Criação de um endpoint VPC para AWS Payment Cryptography

Você pode criar um endpoint de VPC para criptografia de AWS pagamento usando o console do Amazon VPC ou a API do Amazon VPC. Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário do AWS PrivateLink .

- Para criar um endpoint VPC para criptografia de AWS pagamento, use os seguintes nomes de serviço:

```
com.amazonaws.region.payment-cryptography.controlplane
```

```
com.amazonaws.region.payment-cryptography.dataplane
```

Por exemplo, na Região Oeste dos EUA (Oregon) (`us-west-2`), os nomes dos serviços seriam:

```
com.amazonaws.us-west-2.payment-cryptography.controlplane
```

```
com.amazonaws.us-west-2.payment-cryptography.dataplane
```

Para facilitar o uso do endpoint da VPC, é possível habilitar um nome de DNS privado para seu endpoint da VPC. Se você selecionar a opção Ativar nome de DNS, o nome de host DNS AWS de criptografia de pagamento padrão será resolvido para seu endpoint VPC. Por exemplo, `https://controlplane.payment-cryptography.us-west-2.amazonaws.com` resolveria para

um endpoint da VPC conectado ao nome de serviço `com.amazonaws.us-west-2.payment-cryptography.controlplane`.

Essa opção facilita usar o endpoint da VPC. Os AWS SDKs AWS CLI usam o nome de host DNS de criptografia de AWS pagamento padrão por padrão, portanto, você não precisa especificar a URL do endpoint da VPC em aplicativos e comandos.

Para mais informações, consulte Acessar um serviço por meio de um endpoint de interface no Guia do AWS PrivateLink .

Conectando-se a um AWS Endpoint VPC de criptografia de pagamento

Você pode se conectar à criptografia AWS de pagamento por meio do endpoint VPC usando um AWS SDK, o ou. AWS CLI Ferramentas da AWS para PowerShell Para especificar o endpoint da VPC, use seu nome de DNS.

Por exemplo, este comando list-keys usa o parâmetro `endpoint-url` para especificar o endpoint da VPC. Para usar um comando como este, substitua o exemplo de ID de endpoint da VPC na sua conta.

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

Se os nomes de host privados tiverem sido ativados ao criar o endpoint da VPC, você não precisa especificar o URL do endpoint da VPC nos comandos de CLI ou na configuração da aplicação. O nome de host DNS de criptografia de AWS pagamento padrão é resolvido para seu endpoint VPC. Os SDKs AWS CLI e usam esse nome de host por padrão, então você pode começar a usar o VPC endpoint para se conectar a um endpoint regional de criptografia de AWS pagamento sem alterar nada em seus scripts e aplicativos.

Para usar nomes de host privados, os atributos`enableDnsHostnames` e `enableDnsSupport` da sua VPC devem ser definidos como `true`. Para definir esses atributos, use a ModifyVpcAttribute operação. Para mais detalhes, consulte Exibir e atualizar atributos DNS para sua VPC no Guia do usuário do Amazon VPC.

Controlar o acesso a um endpoint da VPC

Para controlar o acesso ao seu endpoint de VPC para criptografia de AWS pagamento, anexe uma política de endpoint de VPC ao seu endpoint de VPC. A política de endpoint determina se os

diretores podem usar o endpoint VPC para chamar operações de criptografia de AWS pagamento com recursos específicos de criptografia de pagamento. AWS

É possível criar uma política de endpoint da VPC ao criar seu endpoint e alterar a política de endpoint da VPC a qualquer momento. Use o console de gerenciamento da VPC ou as ModifyVpcEndpoint operações CreateVpcEndpoint ou. Você também pode criar e alterar uma política de endpoint de VPC usando um AWS CloudFormation modelo. Para obter ajuda sobre o uso do console de gerenciamento da VPC, consulte Criar um endpoint de interface e Modificar um endpoint de interface no Guia do AWS PrivateLink .

Para ajuda sobre como escrever e formatar um documento de política JSON, consulte aReferência a políticas JSON do IAM, no Manual do usuário do IAM.

Tópicos

- Sobre políticas de endpoint da VPC
- Política de endpoint da VPC padrão
- Criar uma política de endpoint da VPC
- Visualizar uma política de endpoint da VPC

Sobre políticas de endpoint da VPC

Para que uma solicitação de criptografia de AWS pagamento que usa um endpoint VPC seja bem-sucedida, o principal exige permissões de duas fontes:

- Uma política baseada em identidade deve dar ao principal permissão para chamar a operação no recurso (chaves de criptografia de AWS pagamento ou alias).
- Uma política de endpoint da VPC deve dar permissão à entidade principal para usar o endpoint para fazer a solicitação.

Por exemplo, uma política de chaves pode dar permissão principal para chamar Decrypt em determinadas chaves de criptografia AWS de pagamento. No entanto, a política de endpoint da VPC pode não permitir que o diretor `Decrypt` invoque essas chaves de criptografia AWS de pagamento usando o endpoint.

Ou uma política de endpoint de VPC pode permitir que um diretor use o endpoint para chamar determinadas chaves de StopKeyUsage criptografia de AWS pagamento. Mas se o diretor não tiver essas permissões de uma política do IAM, a solicitação falhará.

Política de endpoint da VPC padrão

Cada endpoint da VPC tem uma política de endpoint da VPC, mas não é necessário especificar a política. Se você não especificar uma política, a política de endpoint padrão permitirá todas as operações por todas as entidades principais em todos os recursos sobre o endpoint.

No entanto, para recursos AWS de criptografia de pagamento, o diretor também deve ter permissão para chamar a operação a partir de uma política do IAM. Portanto, na prática, a política padrão diz que se uma entidade principal tem permissão para chamar uma operação em um recurso, ela também pode chamá-la usando o endpoint.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Principal": "*",
      "Resource": "*"
    }
  ]
}
```

Para permitir que entidades principais usem o endpoint da VPC apenas para um subconjunto de suas operações permitidas, crie ou atualize a política de endpoint da VPC.

Criar uma política de endpoint da VPC

Uma política de endpoint da VPC determina se uma entidade principal tem permissão para usar o endpoint da VPC para executar operações em um recurso. Para recursos AWS de criptografia de pagamento, o diretor também deve ter permissão para realizar as operações a partir de uma política do IAM.

Cada declaração de política de endpoint da VPC requer os seguintes elementos:

- A entidade principal que pode executar ações
- As ações que podem ser executadas
- Os recursos nos quais as ações podem ser executadas

A declaração de política não especifica o endpoint da VPC. Em vez disso, ele se aplica a qualquer endpoint da VPC ao qual a política está associada. Para obter mais informações, consulte Controlar o acesso a serviços com endpoint da VPCs no Guia do usuário da Amazon VPC.

Veja a seguir um exemplo de uma política de endpoint de VPC para criptografia de AWS pagamentos. Quando anexada a um VPC endpoint, essa política permite usar o VPC endpoint `ExampleUser` para chamar as operações especificadas nas chaves de criptografia de pagamento AWS especificadas. Antes de usar uma política como essa, substitua o exemplo de identificador principal e chave por valores válidos da sua conta.

```
{
   "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 registra todas as operações que usam o endpoint VPC. No entanto, seus CloudTrail registros não incluem operações solicitadas por diretores em outras contas ou operações para chaves de criptografia AWS de pagamento em outras contas.

Dessa forma, talvez você queira criar uma política de endpoint de VPC que impeça que diretores em contas externas usem o endpoint VPC para chamar qualquer operação de criptografia de AWS pagamento em qualquer chave na conta local.

O exemplo a seguir usa a chave de condição PrincipalAccount global aws: para negar acesso a todos os principais para todas as operações em todas as chaves de criptografia de AWS pagamento, a menos que o principal esteja na conta local. Antes de usar uma política como esta, substitua o ID de conta demonstrativo por um válido.

```
{
  "Statement": [
    {
      "Sid": "AccessForASpecificAccount",
      "Principal": {"AWS": "*"},
      "Action": "payment-cryptography:*",
      "Effect": "Deny",
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": "111122223333"
        }
      }
    }
  ]
}
```

Visualizar uma política de endpoint da VPC

Para ver a política de endpoint da VPC para um endpoint, use o console de gerenciamento da VPC ou a operação. DescribeVpcEndpoints

O AWS CLI comando a seguir obtém a política para o endpoint com o ID de endpoint VPC especificado.

Antes de executar esse comando, substitua o ID de endpoint demonstrativo por um válido da sua conta.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==`vpce-1234abcdf5678c90a`].[PolicyDocument]'
--output text
```

Usar um endpoint da VPC em uma declaração de política

Você pode controlar o acesso aos recursos e operações de criptografia de AWS pagamento quando a solicitação vem da VPC ou usa um endpoint da VPC. Para fazer isso, use uma política de IAM

- Use a chave de condição `aws:sourceVpce` para conceder ou restringir o acesso com base no endpoint da VPC.
- Use a chave de condição `aws:sourceVpc` para conceder ou restringir o acesso a uma VPC que hospedar o endpoint privado.

> **Note**
>
> A chave de `aws:sourceIP` condição não é efetiva quando a solicitação vem de um endpoint do Amazon VPC. Para restringir solicitações a um endpoint da VPC, use as chaves de condições `aws:sourceVpce` ou `aws:sourceVpc`. Para obter mais informações, consulte Gerenciamento de identidade e acesso para VPC endpoints e serviços de VPC endpoint no Guia do AWS PrivateLink .

Você pode usar essas chaves de condição globais para controlar o acesso às chaves de criptografia de AWS pagamento, aliases e operações como essas CreateKey que não dependem de nenhum recurso específico.

Por exemplo, o exemplo de política de chaves a seguir permite que um usuário realize operações criptográficas específicas com chaves de criptografia de AWS pagamento somente quando a solicitação usa o endpoint VPC especificado, bloqueando o acesso da Internet e das AWS PrivateLink conexões (se configurado). Quando um usuário faz uma solicitação à criptografia de AWS pagamento, o ID do endpoint da VPC na solicitação é comparado ao valor da chave de `aws:sourceVpce` condição na política. Se não coincidirem, a solicitação será negada.

Para usar uma política como essa, substitua o Conta da AWS ID do espaço reservado e os IDs do endpoint da VPC por valores válidos para sua conta.

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"
                }
            }
        }

    ]
}
```

Você também pode usar a chave de `aws:sourceVpc` condição para restringir o acesso às suas chaves de criptografia AWS de pagamento com base na VPC na qual o endpoint da VPC reside.

O exemplo de política de chaves a seguir permite comandos que gerenciam as chaves de criptografia de AWS pagamento somente quando elas vêm de `vpc-12345678`. Além disso, ele permite comandos que usam as chaves AWS de criptografia de pagamento para operações criptográficas somente quando elas vêm. `vpc-2b2b2b2b` Você pode usar uma política como essa se uma aplicação é executada em uma VPC, mas você usa uma segunda VPC isolada para funções de gerenciamento.

Para usar uma política como essa, substitua o Conta da AWS ID do espaço reservado e os IDs do endpoint da VPC por valores válidos para sua conta.

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": "*"
        }
    ]
}
```

Registrar o endpoint da VPC em log

AWS CloudTrail registra todas as operações que usam o endpoint VPC. Quando uma solicitação para AWS Payment Cryptography usa um endpoint VPC, o ID do endpoint VPC aparece na entrada de AWS CloudTrail registro que registra a solicitação. Você pode usar o ID do endpoint para auditar o uso do seu endpoint VPC de criptografia de AWS pagamento.

Para proteger sua VPC, as solicitações negadas por uma política de endpoint da VPC, mas que de outra forma seriam permitidas, não são registradas. AWS CloudTrail

Por exemplo, este exemplo de entrada de log registra uma solicitação de GenerateMac que usou o VPC endpoint. O campo `vpcEndpointId` aparece no final da entrada de log.

```
{
      "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",
```

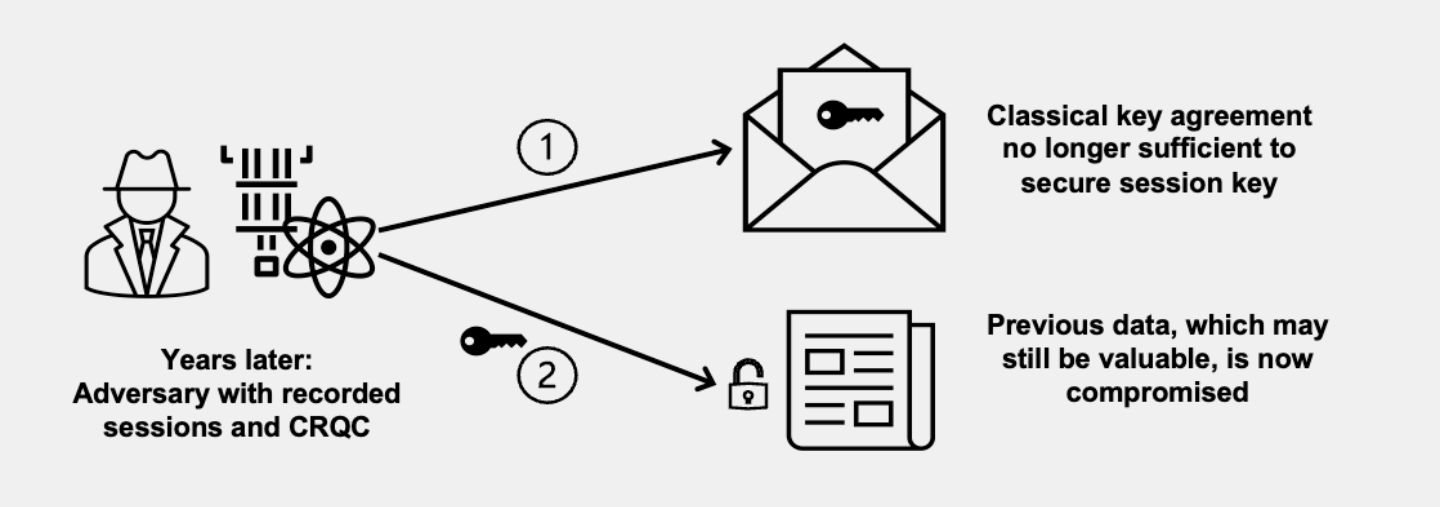
```
                "mfaAuthenticated": "false"
            },
            "ec2RoleDelivery": "2.0"
        }
    },
    "eventTime": "2024-05-27T19:49:54Z",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventName": "CreateKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "172.31.85.253",
    "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"
    }
}
```


Usar TLS pós-quântico híbrido

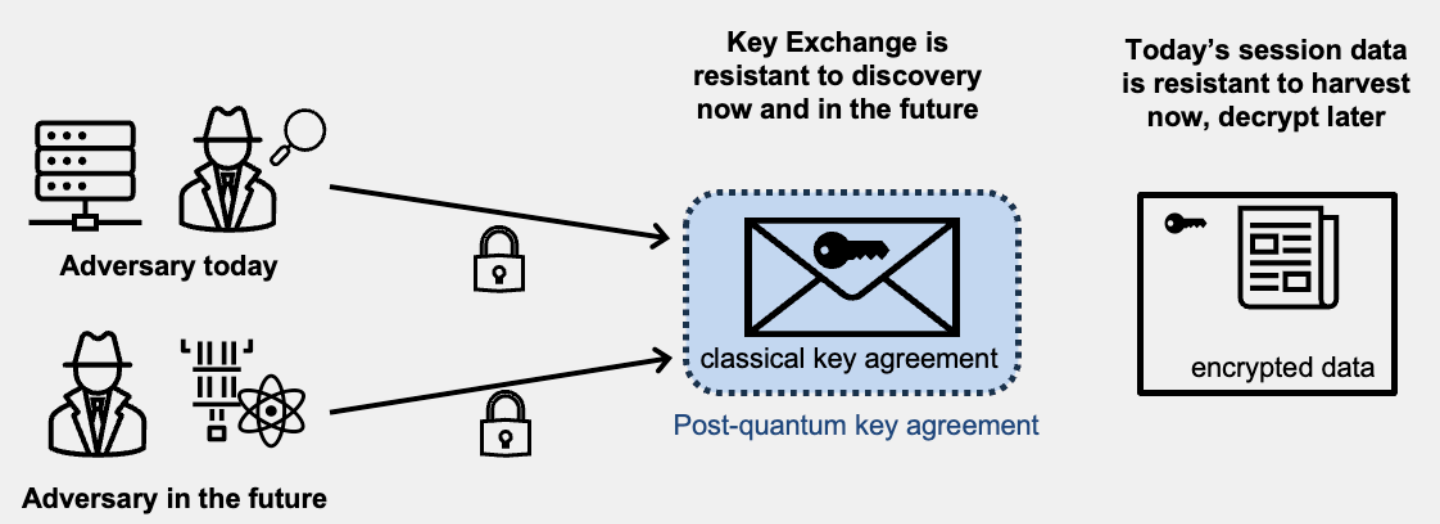
AWS A criptografia de pagamento e muitos outros serviços oferecem suporte a uma opção híbrida de troca de chaves pós-quânticas para o protocolo de criptografia de rede Transport Layer Security (TLS). Você pode usar essa opção de TLS ao se conectar aos endpoints da API ou ao usar os AWS SDKs. Esses recursos opcionais de troca de chaves pós-quânticas híbridas são, no mínimo, tão seguros quanto a criptografia TLS que usamos atualmente e provavelmente fornecerão benefícios adicionais de segurança em longo prazo.

Os dados que você envia para os serviços habilitados são protegidos em trânsito pela criptografia fornecida por uma conexão TLS (Transport Layer Security). Os conjuntos clássicos de criptografia baseados em RSA e ECC que a criptografia de AWS pagamento suporta para sessões de TLS tornam os ataques de força bruta aos mecanismos de troca de chaves inviáveis com a tecnologia atual. No entanto, se computadores quânticos de grande escala ou criptograficamente relevantes (CRQC) se tornarem práticos no futuro, os mecanismos de troca de chaves TLS existentes estarão suscetíveis a esses ataques. É possível que os adversários comecem a coletar dados criptografados agora com a esperança de que possam descriptografá-los no futuro (colha agora, descriptografe mais tarde). Se você estiver desenvolvendo aplicativos que dependem da confidencialidade de longo prazo dos dados transmitidos por uma conexão TLS, considere um plano para migrar para a criptografia pós-quântica antes que computadores quânticos de grande escala estejam disponíveis para uso. AWS está trabalhando para se preparar para esse futuro, e queremos que você também esteja bem preparado.



Para proteger os dados criptografados hoje contra possíveis ataques futuros, AWS está participando com a comunidade criptográfica no desenvolvimento de algoritmos quânticos ou pós-quânticos. AWS implementou pacotes híbridos de criptografia pós-quântica de troca de chaves que combinam elementos clássicos e pós-quânticos para garantir que sua conexão TLS seja pelo menos tão forte quanto seria com os conjuntos de cifras clássicos.

Esses pacotes de criptografia híbrida estão disponíveis para uso em suas cargas de trabalho de produção ao usar versões recentes dos SDKs. AWS Para obter mais informações sobre como fazer enable/disable esse comportamento, consulte Habilitando o TLS híbrido pós-quântico.



Sobre a troca de chaves pós-quântica híbrida no TLS

Os algoritmos usados são um híbrido que combina a curva elíptica Diffie-Hellman (ECDH), um algoritmo clássico de troca de chaves usado atualmente em TLS, com o Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM), um algoritmo de criptografia de chave pública e estabelecimento de chaves que o Instituto Nacional de Padrões e Tecnologia (NIST) designou como seu primeiro algoritmo padrão de acordo de chave pós-quântico. AWS Esse híbrido utiliza cada um dos algoritmos de maneira independente para gerar uma chave. Depois, ele combina as duas chaves criptograficamente.

Como usar

Para obter informações sobre como ativar o TLS pós-quântico híbrido nos AWS SDKs, consulte Habilitando o TLS pós-quântico híbrido.

Saiba mais sobre o PQC

Para obter informações sobre o projeto de criptografia pós-quântica no Instituto Nacional de Padrões e Tecnologia (NIST), consulte Criptografia. Post-Quantum

Para obter informações sobre a padronização de criptografia pós-quântica do NIST, consulte Padronização de criptografia. Post-Quantum

Habilitando o TLS híbrido pós-quântico

Este tópico foi movido para o Guia de referência de AWS SDKs e ferramentas.

Práticas recomendadas de segurança para AWS Payment Cryptography

AWS A criptografia de pagamento oferece suporte a muitos recursos de segurança incorporados ou que você pode implementar opcionalmente para aprimorar a proteção de suas chaves de criptografia e garantir que elas sejam usadas para a finalidade pretendida, incluindo políticas do IAM, um amplo conjunto de chaves de condição da política para refinar suas políticas de chaves e políticas do IAM e a aplicação integrada das regras de PIN do PCI em relação aos blocos de chaves.

> **Important**
>
> As diretrizes gerais fornecidas não representam uma solução de segurança completa. Como nem todas as melhores práticas são apropriadas para todas as situações, elas não se destinam a ser prescritivas.

- Uso de chaves e modos de uso: A criptografia de AWS pagamento segue e impõe restrições de uso e modo de uso de chaves, conforme descrito na especificação de bloco de chaves de troca segura interoperável de chaves ANSI X9 TR 31-2018 e consistente com o requisito de segurança PCI PIN 18-3. Isso limita a capacidade de usar uma única chave para várias finalidades e vincula criptograficamente os metadados da chave (como operações permitidas) ao próprio material da chave. AWS A criptografia de pagamento impõe automaticamente essas restrições, como a de que uma chave de criptografia (TR31_K0_KEY_ENCRYPTION_KEY) também não possa ser usada para decodificação de dados. Consulte Compreendendo os principais atributos do AWS Chave de criptografia de pagamento para obter mais detalhes.
- Limite o compartilhamento de material de chave simétrica: compartilhe material de chave simétrica (como chaves de criptografia PIN ou chaves de criptografia de chave) com, no máximo, uma outra entidade. Se houver a necessidade de transferir material confidencial para mais entidades ou parceiros, crie chaves adicionais. AWS A criptografia de pagamento nunca expõe o material de chave simétrica ou o material de chave privada assimétrica de forma clara.
- Use aliases ou tags para associar chaves a determinados casos de uso ou parceiros: os aliases podem ser usados para indicar facilmente o caso de uso associado a uma chave, como alias/BIN_12345_CVK para indicar uma chave de verificação de cartão associada ao BIN 12345. Para fornecer mais flexibilidade, considere criar tags como bin=12345, use_case=acquiring,country=us,partner=foo. Aliases e tags também podem ser usados para limitar o acesso, como impor controles de acesso entre a emissão e a aquisição de casos de uso.

- Pratique o acesso com privilégio mínimo: o IAM pode ser usado para limitar o acesso de produção a sistemas em vez de indivíduos, como proibir usuários individuais de criar chaves ou executar operações criptográficas. O IAM também pode ser usado para limitar o acesso a comandos e chaves que podem não ser aplicáveis ao seu caso de uso, como limitar a capacidade de gerar ou validar PINs para um adquirente. Outra forma de usar o acesso com privilégio mínimo é restringir operações confidenciais (como importação de chaves) a contas de serviço específicas. Consulte AWS Exemplos de políticas baseadas em identidade de criptografia de pagamento para ver exemplos.

Consulte também

- Gerenciamento de identidade e acesso para AWS Payment Cryptography
- Práticas recomendadas de segurança no IAM, noManual do usuário do IAM

Validação de conformidade para criptografia AWS de pagamento

Assim como em outros AWS serviços, os clientes precisam de uma compreensão clara do modelo de responsabilidade compartilhada para segurança e conformidade. Como um serviço que oferece suporte específico a pagamentos, é particularmente importante entender a conformidade com os padrões PCI aplicáveis para clientes de criptografia AWS de pagamento. AWS As avaliações do PCI DSS e do PCI 3DS incluem criptografia de pagamento. AWS Pode haver referências ao serviço nos Guias de Responsabilidade Compartilhada, disponíveis em AWS Artifact, para esses relatórios. As avaliações de segurança e Point-to-Point criptografia de PIN PCI (P2PE) são específicas para criptografia de pagamento. AWS

Esta seção fornece informações sobre o status e o escopo da conformidade do serviço e informações que serão úteis no planejamento das avaliações de segurança PCI PIN e PCI P2PE de seus aplicativos.

Tópicos

- Conformidade do serviço
- Planejamento de conformidade com PIN
- Usando o componente AWS de decodificação de criptografia de pagamento em soluções P2PE

Conformidade do serviço

Auditores terceirizados avaliam a segurança e a conformidade da criptografia de AWS pagamento como parte de vários programas de AWS conformidade. Isso inclui SOC, PCI e outros.

AWS A criptografia de pagamento foi avaliada para vários padrões PCI, além do PCI DSS e do PCI 3DS. Isso inclui segurança PCI PIN (PCI PIN) e criptografia PCI Point-to-Point (P2PE). Consulte AWS Artifact os atestados e guias de conformidade disponíveis.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte Serviços da AWS no escopo do programa de conformidade . Para obter informações gerais, consulte Programas de conformidade da AWS.

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Sua responsabilidade de conformidade ao usar a criptografia de AWS pagamento é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade da sua empresa e pelas leis e regulamentos aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido de segurança e compatibilidade: esses guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- AWS Recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- Avaliar recursos com regras no Guia do desenvolvedor do AWS Config :AWS Config; avalia como suas configurações de recursos estão em conformidade com práticas internas, diretrizes e regulamentos do setor.
- AWS Security Hub CSPM—Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Planejamento de conformidade com PIN

Este guia descreve a documentação e as evidências de que você precisará se preparar para uma avaliação do PIN PCI do seu aplicativo de processamento de PIN que usa criptografia AWS de pagamento.

Assim como acontece com outros Serviços da AWS padrões de conformidade, é sua responsabilidade usar o serviço com segurança, configurando o controle de acesso e usando parâmetros de segurança alinhados aos requisitos de PIN do PCI. Este guia discutirá essas configurações quando apropriado para atender a um requisito.

Tópicos

- Tópicos comuns
- Escopo da avaliação
- Operações de processamento de transações

Tópicos comuns

A migração de aplicativos da conexão com o HSM para um serviço gerenciado, como criptografia de AWS pagamento, traz à tona problemas e conceitos comuns para clientes e seus avaliadores. Esta seção fornece informações para esclarecer como o uso seguro do serviço aborda essas situações.

Tópicos

- Responsabilidade compartilhada
- Configuração mínima do HSM
- Troca de chaves entre o cliente e a APC

Responsabilidade compartilhada

Os clientes que assumiram total responsabilidade pela segurança e conformidade pelos aplicativos reestruturarão sua conformidade para aproveitar as vantagens do gerenciamento de chaves, dos controles de segurança e dos recursos gerenciados de HSM (“o serviço”) da AWS Payment Cryptography. Isso mudará completamente alguns requisitos para AWS, conforme atestado pelas avaliações de terceiros da AWS Payment Cryptography. Alguns requisitos serão compartilhados entre o aplicativo do cliente e o serviço. Um aplicativo é responsável por:

- Fornecendo informações precisas ao serviço
- Usando controles de segurança de acordo com as recomendações do serviço e os requisitos de segurança do PCI PIN
- Implementando os controles de segurança necessários usando as ferramentas fornecidas pelo serviço

Os clientes e seus avaliadores usarão guias de implementação e responsabilidade compartilhada publicados com atestados de conformidade AWS Artifact para implementar controles e monitoramento de controle e, em seguida, planejar e concluir as avaliações.

Configuração mínima do HSM

O Padrão de Segurança de Dados PCI, o padrão fundamental para outros padrões PCI, exige que todos os sistemas sejam configurados com a funcionalidade mínima necessária para sua função. PCI PIN, P2PE e outros padrões de solução aplicam esse requisito HSMs na solução. HSMs só deve habilitar as funções necessárias para a solução.

AWS os serviços devem ser tratados como sistemas e configurados para a funcionalidade mínima necessária. O Payment Card Industry Data Security Standard (PCI DSS) v4.0 na AWS recomenda o uso do IAM para configurar a funcionalidade mínima para cada serviço da AWS usado pela solução. Isso também se aplica à criptografia AWS de pagamento. As políticas do IAM permitem permissões refinadas para restringir as funções criptográficas somente aos componentes do aplicativo que dependem delas.

Troca de chaves entre o cliente e a APC

PIN PIN Os requisitos de segurança 8-4 e 15-2 exigem que as chaves públicas para troca e carregamento de chaves sejam autenticadas e protegidas pela integridade. Para o carregamento remoto por chave de POI, funcionalmente descrito no ANSI/ASC X9 TR-34 e regido pelo PIN PCI Anexo A, as chaves públicas são geralmente transmitidas em certificados assinados por uma autoridade de certificação compatível com o Anexo A2. Para trocas entre organizações, as chaves públicas usam outros mecanismos de autenticidade e integridade.

Todas as interações entre o cliente e a AWS são feitas via AWS APIs, que autentica mutuamente cada chamada de API e garante a integridade das chamadas e respostas usando o TLS. A autenticação do aplicativo do cliente é gerenciada pelo AWS Identity and Access Management com mecanismos como Security Tokens e SigV4. Os endpoints da API da AWS são autenticados pelo cliente usando a autenticação do servidor TLS, que é incorporada à AWS. SDKs Então, o TLS garante a confidencialidade e a integridade de todos os dados transmitidos entre o cliente e cada API da AWS.

APC APIs GetParametersForImport e ImportKey implemente uma transferência de chaves do cliente para o serviço. Embora a Autoridade Certificadora (CA) fornecida por não GetParametersForImport esteja em conformidade com o Anexo A2, ela é segura e exclusiva para a conta. Embora essa CA não seja confiável para conformidade com os requisitos 8-4 e 15-2, ela fornece verificação de integridade da chave importada. Você também pode usar sua própria CA aproveitando a GetCertificateSigningRequest API.

Os mecanismos que fornecem autenticação de chave pública e garantia de integridade são:

- Autenticação fornecida pela AWS API Authentication
- A integridade da chave é fornecida pelo recurso MAC do certificado fornecido por GetParametersForImport, mesmo que as informações de identidade no certificado não sejam confiáveis. A integridade da chave também é garantida pelo MAC usado pelo TLS, protegendo a sessão entre o cliente e a AWS.

Os certificados e blocos de chaves fornecidos pela APC estão em conformidade com o Anexo A1, que especifica os requisitos para certificados e proteção de chaves por métodos assimétricos.

Escopo da avaliação

A primeira etapa no planejamento de qualquer avaliação é documentar o escopo. Para o PCI PIN, o escopo são sistemas e processos que protegem os PINs, incluindo a proteção das chaves criptográficas e dos dispositivos que os protegem — terminais de pagamento, também chamados de pontos de interação (POI), HSMs e outros dispositivos criptográficos seguros (SCD).

Não abordaremos os requisitos nos quais você assume total responsabilidade, pois eles abordam áreas fora do escopo do serviço. Por exemplo, configuração e provisionamento de terminais de pagamento. Consulte o Guia de Responsabilidade Compartilhada por Criptografia de AWS Pagamento para PIN PCI, disponível em AWS Artifact

Tópicos

- Responsabilidade compartilhada
- High-Level Diagramas de rede
- Tabela chave
- Referências de documentos

Responsabilidade compartilhada

AWS A Payment Cryptography é uma Organização de Encriptação e Suporte (ESO) e uma PIN-Acquiring Third-Party Prestadora de Serviços (TPS), conforme definida pelo Programa de Segurança de PIN da Visa e listada no Registro de Provedores de Serviços Globais da Visa, em “Amazon Web Services, LLC”. Isso significa que a Visa permite que o serviço seja usado pelo PIN-Acquiring Third-Party VisaNet Processador (VNP), pelo Processador do PIN-Acquiring Cliente VisaNet atuando como Provedor de Serviços e por outros provedores de TPS e ESO sem exigir uma avaliação adicional por parte dos avaliadores de PIN do cliente (avaliadores de PIN qualificados pelo PCI ou PCI QPA).

Outras marcas de cartões ou provedores de redes de pagamento podem confiar no Programa de Segurança de PIN da Visa ou ter seus próprios programas. Entre em contato AWS Support se tiver dúvidas sobre a conformidade dos serviços de outros programas de rede de pagamento.

AWS fornece o atestado de conformidade de segurança PCI PIN (AOC) e o Guia de Responsabilidade Compartilhada para AWS criptografia de pagamentos em. AWS Artifact O uso de provedores de serviços no processamento de PIN é comum há muitos anos, no entanto, o Padrão

de Segurança PCI PIN, até a versão 3.1, não aborda o gerenciamento de provedores de serviços terceirizados. Nem o Programa de Segurança de PIN da Visa. O QPA do cliente seguiu o modelo estabelecido com o AOC do PCI DSS e o Guia de Responsabilidade Compartilhada de se referir à AWS“conformidade” como bem-sucedida no teste dos requisitos aplicáveis.

High-Level Diagramas de rede

O modelo de relatório de PIN do PCI exige: “Para entidades envolvidas no processamento de transações baseadas em PIN, forneça um esquema de rede descrevendo fluxos de transações baseados em PIN com o uso do tipo de chave associado. Além disso, os KIFs e entidades envolvidas na distribuição remota de chaves usando técnicas assimétricas devem fornecer fluxos de material chave.”

AWS A Payment Cryptography relatou a estrutura interna do serviço para nossa avaliação de PIN PCI. Seus diagramas ilustrarão a chamada das APIs de serviço para processamento de PIN.

Exemplo de diagrama de rede de alto nível para aplicativos de PIN usando criptografia AWS de pagamento:

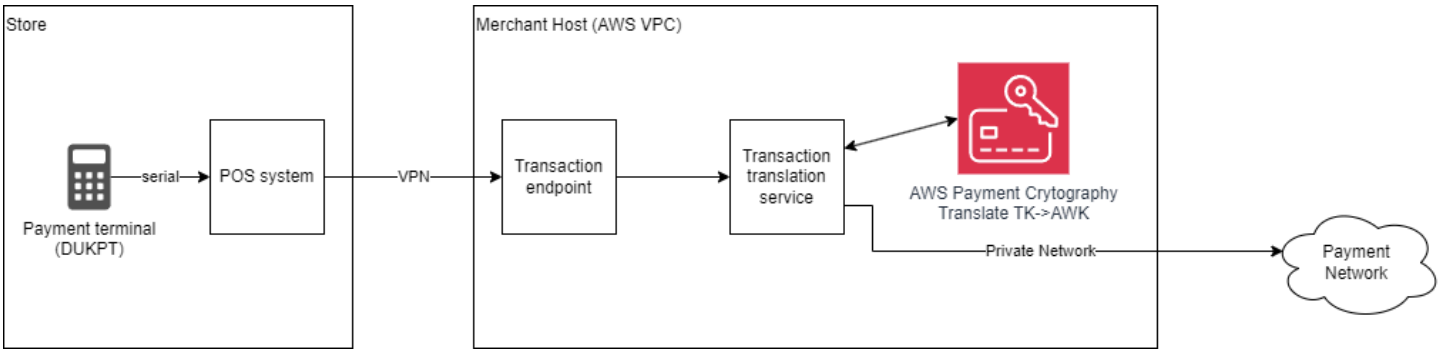


Tabela chave

O relatório exige que todas as chaves que protegem os PINs, direta ou indiretamente, sejam listadas. Todas as chaves que existem no serviço podem ser listadas com a ListKeysAPI.

Certifique-se de fornecer a lista de chaves para todas as regiões e contas que possuem chaves para seu aplicativo.

Referências de documentos

A documentação e as recomendações do fornecedor para o uso seguro da criptografia de AWS pagamento estão no Guia do Usuário e na Referência da API. Eles estão vinculados, conforme apropriado, nesta orientação.

Operações de processamento de transações

Os requisitos de PIN do PCI são organizados em Objetivos de Controle. Cada objetivo de controle agrupa os requisitos para garantir um aspecto da segurança dos PINs.

Tópicos

- Objetivo de controle 1: Os PINs usados em transações regidas por esses requisitos são processados usando equipamentos e metodologias que garantem que sejam mantidos em segurança.
- Objetivo de controle 2: As chaves criptográficas usadas para o PIN encryption/decryption e o gerenciamento de chaves relacionadas são criadas usando processos que garantem que não seja possível prever nenhuma chave ou determinar que certas chaves são mais prováveis do que outras chaves.
- Objetivo de controle 3: As chaves são transportadas ou transmitidas de maneira segura.
- Objetivo de controle 4: Key-loading os HSMs e PIN-acceptance dispositivos POI são tratados de forma segura.
- Objetivo de controle 5: As chaves são usadas de forma a impedir ou detectar seu uso não autorizado.
- Objetivo de controle 6: As chaves são administradas de forma segura.
- Objetivo de controle 7: O equipamento usado para processar PINs e chaves é gerenciado de maneira segura.

Objetivo de controle 1: Os PINs usados em transações regidas por esses requisitos são processados usando equipamentos e metodologias que garantem que sejam mantidos em segurança.

Requisito 1: Os HSMs usados pela criptografia AWS de pagamento foram avaliados como parte de nossa avaliação de PIN PCI. Para clientes que usam o serviço, os requisitos 1-3 e 1-4 estão “em vigor” em relação ao HSM gerenciado pelo serviço. As descobertas do HSM indicarão que o teste foi atestado pelo AWS QPA. O Atestado de Conformidade do PIN está disponível para ser referenciado em. AWS Artifact Outros SCD, como POI, em sua solução precisarão ser inventariados e referenciados.

Requisito 2: A documentação de seus procedimentos deve especificar como os PINs do titular do cartão são protegidos em relação à divulgação à sua equipe, aos protocolos de tradução de

PIN implementados e à proteção durante o processamento on-line e off-line. Além disso, sua documentação deve conter um resumo dos métodos de gerenciamento de chaves criptográficas usados em cada zona.

Requisito 3: O POI deve ser configurado para criptografia e transmissão seguras de PIN. AWS A criptografia de pagamento suporta somente traduções de blocos de PIN especificadas no Requisito 3-3.

Requisito 4: O aplicativo não deve armazenar blocos de PIN. Os blocos de PIN, mesmo criptografados, não devem ser retidos em diários ou registros de transações. O serviço não armazena blocos de PIN e a avaliação de PIN verifica se eles não estão nos registros.

Observe que o padrão PCI PIN Security se aplica à aquisição “do gerenciamento, processamento e transmissão seguros de dados do número de identificação pessoal (PIN) durante o processamento online e offline de transações com cartões de pagamento em caixas eletrônicos e terminais de ponto de venda (POS)”, conforme declarado no padrão. No entanto, o padrão é frequentemente usado para avaliar o gerenciamento de chaves criptográficas para pagamentos fora do escopo pretendido. Isso pode incluir casos de uso do emissor em que os PINs são armazenados. As exceções aos requisitos para esses casos devem ser acordadas com o público-alvo da avaliação.

Objetivo de controle 2: As chaves criptográficas usadas para o PIN encryption/decryption e o gerenciamento de chaves relacionadas são criadas usando processos que garantem que não seja possível prever nenhuma chave ou determinar que certas chaves são mais prováveis do que outras chaves.

Requisito 5: A geração de chaves por criptografia de AWS pagamento foi avaliada como parte de nossa avaliação de PIN PCI. Isso pode ser especificado na coluna “Gerado por” da tabela de chaves.

Requisito 6: Os controles de segurança das chaves mantidas na criptografia de AWS pagamento foram avaliados como parte da avaliação do PIN PCI do serviço. Inclua descrições dos controles de segurança relacionados à geração de chaves em seu aplicativo e com qualquer outro provedor de serviços.

Requisito 7: Você deve ter uma documentação de política de geração de chaves que especifique como as chaves são geradas e todas as partes afetadas devem estar cientes delas. procedures/policies Os procedimentos para criação de chaves usando a API da APC devem incluir o uso de funções com permissões de criação de chaves e aprovações para executar scripts ou outro código que crie chaves. AWS CloudTrail os registros contêm todos os CreateKeyeventos com data e hora,

ARN da chave e IDs de usuário. Os números de série e registros do HSM para acesso à mídia física foram avaliados como parte da avaliação do PIN do serviço.

Objetivo de controle 3: As chaves são transportadas ou transmitidas de maneira segura.

Requisito 8: A transferência de chaves com criptografia de AWS pagamento foi avaliada como parte de nossa avaliação de PIN PCI. Você precisará documentar os principais mecanismos de proteção das transferências antes da importação e após a exportação da Criptografia de AWS Pagamento. O serviço fornece valores de verificação de chaves para todas as chaves para validar o transporte correto.

O requisito 8-4 exige que as chaves públicas sejam transmitidas de forma a proteger sua integridade e autenticidade. A transmissão entre seu aplicativo e AWS é controlada pela autenticação do aplicativo para AWS, usando AWS Identity and Access Management métodos, AWS a autenticação de ponto final da API para o aplicativo por meio de certificados de servidor TLS. Além disso, as chaves públicas exportadas ou importadas para a Criptografia AWS de Pagamento têm certificados assinados por CAs efêmeras e específicas do cliente (consulte, e). GetPublicKeyCertificateGetParametersForImportGetParametersForExport Essas CAs não podem ser usadas como o único método de autenticação, porque não são compatíveis com o Anexo A2 de Segurança PCI PIN. No entanto, os certificados ainda fornecem garantia de integridade para chaves públicas com o IAM fornecendo autenticação.

Ao trocar chaves públicas com seus parceiros de negócios usando métodos assimétricos, você deve fornecer a autenticação da empresa por meio do canal de comunicação, usando um site seguro de troca de arquivos, por exemplo.

Requisito 9: O serviço não usa nem oferece suporte direto aos principais componentes de texto não criptografado.

Requisito 10: O serviço impõe a força relativa das chaves de proteção para transporte. Você é responsável pela transferência da chave antes da importação para e depois da exportação da AWS Payment Cryptography e pelo uso de API e TR-31 parâmetros precisos para importação, exportação e geração de chaves. Você deve ter procedimentos documentados para descrever os mecanismos de transporte de chaves e a lista de chaves criptográficas usadas para o transporte.

Requisito 11: A documentação de seus procedimentos deve especificar como as chaves são transmitidas. Os procedimentos para transferência de chaves usando a API AWS Payment Cryptography devem incluir o uso de funções com permissões de importação e exportação de

chaves e aprovações para executar scripts ou outros códigos que criem chaves. AWS CloudTrail os registros contêm tudo ImportKeye ExportKeyeventos.

Objetivo de controle 4: Key-loading os HSMs e PIN-acceptance dispositivos POI são tratados de forma segura.

Requisito 12: Você é responsável por carregar as chaves dos componentes ou compartilhamentos. O gerenciamento das chaves principais do HSM foi avaliado como parte da avaliação do PIN do serviço. AWS A criptografia de pagamento não carrega chaves de ações ou componentes individuais. Consulte a seção Detalhes criptográficos.

Requisitos 13 e 14: Você precisará descrever a proteção das chaves para transferências antes da importação e após a exportação do serviço.

Requisito 15: A criptografia de AWS pagamento fornece valores-chave de verificação para todas as chaves no serviço e garantia de integridade para chaves públicas. Seu aplicativo é responsável por usar essas verificações para validar as chaves após a importação ou exportação do serviço. Você deve documentar os procedimentos para garantir a existência de um mecanismo de validação.

O requisito 15-2 exige que as chaves públicas sejam carregadas de forma a proteger sua integridade e autenticidade. ImportKey, junto com GetParametersForImport, fornece a validação dos certificados de assinatura fornecidos. Se os certificados fornecidos forem autoassinados, a autenticação deverá ser fornecida por um mecanismo separado, por exemplo, troca segura de arquivos.

Requisito 16: A documentação de seus procedimentos deve especificar como as chaves são carregadas no serviço. Os procedimentos para importação de chaves usando a API devem incluir o uso de funções com permissões de importação de chaves e aprovações para executar scripts ou outros códigos que carreguem chaves. AWS CloudTrail os registros contêm todos os ImportKeyeventos. Você deve incluir os mecanismos de registro na documentação. O serviço fornece valores de verificação de chave para todas as chaves para validar o carregamento correto da chave.

Objetivo de controle 5: As chaves são usadas de forma a impedir ou detectar seu uso não autorizado.

Requisito 17: O serviço fornece mecanismos, como tags e aliases, para chaves que permitem o rastreamento de relacionamentos de compartilhamento de chaves. Além disso, os valores de verificação de chave devem ser mantidos separadamente para demonstrar que valores de chave conhecidos ou padrão não são usados quando as chaves são compartilhadas.

Requisito 18: O serviço fornece verificações de integridade de chaves, via GetKeye ListKeys, e eventos de gerenciamento de chaves, via AWS CloudTrail, que podem ser usados para detectar substituições não autorizadas ou monitorar a sincronização de chaves entre as partes. O serviço armazena as chaves exclusivamente em blocos de chaves. Você é responsável pelo armazenamento e uso das chaves antes da importação e após a exportação da Criptografia de AWS Pagamento.

Você deve ter procedimentos para uma investigação imediata caso ocorra alguma discrepância durante o processamento de transações baseadas em PIN ou eventos inesperados de gerenciamento de chaves.

Requisito 19: O serviço usa chaves exclusivamente em blocos de chaves KeyUsage KeyModeOfUse, imposição e outros atributos-chave para todas as operações. Isso inclui restrições às operações de chave privada. Você deve usar suas chaves públicas para uma única finalidade, por exemplo, criptografia ou verificação de assinatura digital, mas não ambas. Você deve usar contas separadas para produção e test/development sistemas.

Requisito 20: Você mantém a responsabilidade por esse requisito.

Objetivo de controle 6: As chaves são administradas de forma segura.

Requisito 21: O armazenamento e o uso de chaves com criptografia de AWS pagamento foram avaliados como parte da avaliação do PIN PCI do serviço. Para requisitos de armazenamento relacionados aos principais componentes, você é responsável por armazená-los conforme descrito em 21-2 e 21-3. Você precisará descrever os principais mecanismos de proteção na documentação da política antes de importar para e depois da exportação do serviço.

Requisito 22: Os principais procedimentos de comprometimento da criptografia de AWS pagamento foram avaliados como parte da avaliação do PIN PCI do serviço. Você precisará descrever os principais procedimentos de detecção e resposta de comprometimentos, incluindo monitoramento e resposta à notificação do AWS.

Requisito 23: A criptografia de AWS pagamento não suporta variantes ou outros métodos reversíveis de cálculo de chave. As chaves principais da APC ou as chaves codificadas por elas nunca estão disponíveis para os clientes. O uso do cálculo de chave reversível foi avaliado como parte da avaliação do PIN do PCI do serviço.

Requisito 24: Práticas de destruição de chaves privadas e secretas internas A criptografia de AWS pagamento foi avaliada como parte da avaliação do PIN PCI do serviço. Você precisará descrever o procedimento de destruição de chaves antes da importação para e após a exportação da APC. Os

requisitos de destruição relacionados aos principais componentes (24-2.2 e 24-2.3) permanecem de sua responsabilidade.

Requisito 25: O acesso a chaves secretas e privadas na criptografia de AWS pagamento foi avaliado como parte da avaliação do PIN PCI do serviço. Você precisará ter um processo e uma documentação para controles de acesso das chaves antes da importação e após a exportação da Criptografia de AWS Pagamento.

Requisito 26: Você precisará descrever o registro de qualquer acesso às chaves, componentes principais ou materiais relacionados usados fora do serviço. Os registros de todas as principais atividades de gerenciamento que seu aplicativo realiza com o serviço estão disponíveis via AWS CloudTrail.

Requisito 27: Você precisará descrever os procedimentos de backup para chaves, componentes principais ou materiais relacionados usados fora do serviço.

Requisito 28: Os procedimentos para toda administração de chaves usando a API devem incluir o uso de funções com permissões de administração de chaves e aprovações para executar scripts ou outro código que gerencie chaves. AWS CloudTrail os registros contêm todos os principais eventos de administração

Objetivo de controle 7: O equipamento usado para processar PINs e chaves é gerenciado de maneira segura.

Requisito 29: Seus requisitos de proteção física e lógica para HSMs são atendidos pelo uso da criptografia de AWS pagamento.

Requisito 30: Seu aplicativo será responsável por toda a proteção física e lógica dos requisitos do dispositivo POI.

Requisito 31: A proteção de dispositivos criptográficos seguros (SCD) usados pela criptografia de AWS pagamento foi avaliada como parte da avaliação do PIN PCI do serviço. Você precisará demonstrar a proteção de quaisquer outros SCDs usados pelo seu aplicativo.

Requisito 32: O uso de SCDs usados pela criptografia AWS de pagamento foi avaliado como parte da avaliação do PIN PCI do serviço. Você precisará demonstrar o controle de acesso e a proteção de quaisquer outros SCDs usados pelo seu aplicativo.

Requisito 33: Você precisará descrever as proteções de qualquer PIN-processing equipamento sob seu controle.

Usando o componente AWS de decodificação de criptografia de pagamento em soluções P2PE

As soluções PCI P2PE podem usar o componente de decodificação AWS de criptografia de pagamento. Isso está documentado na Point-to-Point Criptografia PCI: requisitos de segurança e procedimentos de teste, seção Soluções P2PE e uso de fornecedores terceirizados de componentes and/or P2PE: “Um provedor de soluções (ou um comerciante como provedor de soluções) pode terceirizar determinadas funções P2PE para fornecedores de componentes P2PE listados no PCI e relatar o uso dos componentes P2PE listados no PCI em seu Relatório de Validação P2PE (P-ROV)”, que está disponível no site do PCI.

Assim como acontece com outros serviços e padrões de conformidade da AWS, é sua responsabilidade usar o serviço com segurança, configurando o controle de acesso e usando parâmetros de segurança alinhados aos requisitos do PCI P2PE. O Guia do usuário do componente de decodificação P2PE de criptografia de pagamento da AWS, que está disponível em AWS Artifact, tem instruções detalhadas para integrar a criptografia de AWS pagamento à sua solução PCI P2PE e o relatório anual do componente de descriptografia, que é necessário para relatórios de conformidade.

Gerenciamento de identidade e acesso para AWS Payment Cryptography

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (tem permissões) para usar os recursos de criptografia AWS de pagamento. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

- Público
- Autenticação com identidades
- Gerenciar o acesso usando políticas
- Como AWS A criptografia de pagamento funciona com o IAM
- AWS Exemplos de políticas baseadas em identidade de criptografia de pagamento
- Resource-based políticas para AWS Payment Cryptography
- Multi-party aprovação para criptografia AWS de pagamento
- Solução de problemas AWS Criptografia de pagamento, identidade e acesso

Público

A forma como você usa AWS Identity and Access Management (IAM) difere com base na sua função:

- Usuário do serviço: solicite permissões ao seu administrador se você não conseguir acessar os atributos (consulte Solução de problemas AWS Criptografia de pagamento, identidade e acesso).
- Administrador do serviço: determine o acesso do usuário e envie solicitações de permissão (consulte Como AWS A criptografia de pagamento funciona com o IAM)
- Administrador do IAM: escreva políticas para gerenciar o acesso (consulte AWS Exemplos de políticas baseadas em identidade de criptografia de pagamento)

Autenticação com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado como usuário do IAM ou assumindo uma função do IAM. Usuário raiz da conta da AWS

Você pode fazer login como uma identidade federada usando credenciais de uma fonte de identidade como Centro de Identidade do AWS IAM (IAM Identity Center), autenticação de login único ou credenciais. Google/Facebook Para ter mais informações sobre como fazer login, consulte Como fazer login em sua Conta da AWS no Guia do usuário do Início de Sessão da AWS .

Para acesso programático, AWS fornece um SDK e uma CLI para assinar solicitações criptograficamente. Para ter mais informações, consulte AWS Signature Version 4 para solicitações de API no Guia do usuário do IAM.

Conta da AWS usuário-raiz

Ao criar um Conta da AWS, você começa com uma identidade de login chamada usuário Conta da AWS raiz que tem acesso completo a todos Serviços da AWS os recursos. É altamente recomendável não usar o usuário-raiz em tarefas diárias. Consulte as tarefas que exigem credenciais de usuário-raiz em Tarefas que exigem credenciais de usuário-raiz no Guia do usuário do IAM.

Usuários e grupos do IAM

Um usuário do IAM é uma identidade com permissões específicas para uma única pessoa ou aplicação. É recomendável usar credenciais temporárias, em vez de usuários do IAM com credenciais de longo prazo. Para obter mais informações, consulte Exigir que usuários humanos usem a federação com um provedor de identidade para acessar AWS usando credenciais temporárias no Guia do usuário do IAM.

Um grupo do IAM especifica um conjunto de usuários do IAM e facilita o gerenciamento de permissões para grandes conjuntos de usuários. Para ter mais informações, consulte Casos de uso de usuários do IAM no Guia do usuário do IAM.

Perfis do IAM

Uma perfil do IAM é uma identidade com permissões específicas que oferece credenciais temporárias. Você pode assumir uma função mudando de um usuário para uma função do IAM (console) ou chamando uma operação de AWS API AWS CLI ou. Para saber mais, consulte Métodos para assumir um perfil no Manual do usuário do IAM.

Os perfis do IAM são úteis para acesso de usuário federado, permissões de usuário do IAM temporárias, acesso entre contas, acesso entre serviços e aplicações em execução no Amazon EC2. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Gerenciar o acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política define permissões quando associada a uma identidade ou recurso. AWS avalia essas políticas quando um diretor faz uma solicitação. A maioria das políticas é armazenada AWS como documentos JSON. Para ter mais informações sobre documentos de política JSON, consulte Visão geral das políticas JSON no Guia do usuário do IAM.

Por meio de políticas, os administradores especificam quem tem acesso a que, definindo qual entidade principal pode realizar ações em quais recursos e sob quais condições.

Por padrão, usuários e perfis não têm permissões. Um administrador do IAM cria políticas do IAM e as adiciona aos perfis, os quais os usuários podem então assumir. As políticas do IAM definem permissões, independentemente do método usado para realizar a operação.

Identity-based políticas

Identity-based políticas são documentos de políticas de permissões JSON que você anexa a uma identidade (usuário, grupo ou função). Essas políticas controlam quais ações as identidades podem realizar, em quais recursos e sob quais condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

Identity-based as políticas podem ser políticas em linha (incorporadas diretamente em uma única identidade) ou políticas gerenciadas (políticas autônomas anexadas a várias identidades). Para saber como escolher entre uma política gerenciada e políticas em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do usuário do IAM.

Resource-based políticas

Resource-based políticas são documentos de política JSON que você anexa a um recurso. Entre os exemplos estão políticas de confiança de perfil do IAM e políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. É necessário especificar uma entidade principal em uma política baseada em recursos.

Resource-based as políticas são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. AWS WAF Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do Desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais que podem definir o máximo de permissões concedidas por tipos de políticas mais comuns:

- Limites de permissões: definem o número máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM. Para saber mais sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do usuário do IAM.
- Políticas de Controle de Serviços (SCPs): as SCPs especificam o número máximo de permissões para uma organização ou uma unidade organizacional no AWS Organizations. Para saber mais, consulte Políticas de controle de serviço no Guia do usuário do AWS Organizations .
- Políticas de controle de recursos (RCPs): definem o número máximo de permissões disponíveis para recursos em suas contas. Consulte mais informações em Resource control policies (RCPs) no Guia do usuário do AWS Organizations .
- Políticas de sessão: políticas avançadas transmitidas como um parâmetro durante a criação de uma sessão temporária para um perfil ou um usuário federado. Para saber mais, consulte Políticas de sessão no Guia do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

Como AWS A criptografia de pagamento funciona com o IAM

Antes de usar o IAM para gerenciar o acesso à criptografia de AWS pagamento, você deve entender quais recursos do IAM estão disponíveis para uso com a criptografia AWS de pagamento. Para ter uma visão geral de como a criptografia de AWS pagamento e outros AWS serviços funcionam com o IAM, consulte AWS Serviços que funcionam com o IAM no Guia do usuário do IAM.

Tópicos

- AWS Políticas de criptografia Identity-based de pagamento
- Autorização baseada em AWS Tags de criptografia de pagamento

AWS Políticas de criptografia Identity-based de pagamento

Com as políticas baseadas em identidade do IAM, você pode especificar ações e recursos permitidos ou negados, bem como as condições sob as quais as ações são permitidas ou negadas. AWS A criptografia de pagamento oferece suporte a ações, recursos e chaves de condição específicos. Para conhecer todos os elementos usados em uma política JSON, consulte Referência de elementos de política JSON do IAM no Guia do usuário do IAM.

Ações

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Action` de uma política JSON descreve as ações que podem ser usadas para permitir ou negar acesso em uma política. Incluem ações em uma política para conceder permissões para executar a operação associada.

As ações de política na criptografia de AWS pagamento usam o seguinte prefixo antes da ação:. `payment-cryptography:` Por exemplo, para conceder permissão a alguém para executar uma operação da API `VerifyCardData` de AWS Payment Cryptography, você inclui a ação `payment-cryptography:VerifyCardData` na política. As instruções de política devem incluir um elemento `Action` ou `NotAction`. AWS A criptografia de pagamento define seu próprio conjunto de ações que descrevem as tarefas que você pode realizar com esse serviço.

Para especificar várias ações em uma única instrução, separe-as com vírgulas, como segue:

```
"Action": [
      "payment-cryptography:action1",
```

```
      "payment-cryptography:action2"
```

Você também pode especificar várias ações usando caracteres curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `List` (como `ListKeys` e `ListAliases`), inclua a seguinte ação:

```
"Action": "payment-cryptography:List*"
```

Para ver uma lista de ações de criptografia de AWS pagamento, consulte Ações definidas pela criptografia AWS de pagamento no Guia do usuário do IAM.

Recursos

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Para ações que não oferecem compatibilidade com permissões em nível de recurso, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

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

O recurso da chave de Payment Cryptography tem o ARN a seguir:

```
arn:${Partition}:payment-cryptography:${Region}:${Account}:key/${keyARN}
```

Para obter mais informações sobre o formato dos ARNs, consulte Amazon Resource Names (ARNs) e AWS Service Namespaces.

Por exemplo, para especificar a instância `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` na instrução, use o seguinte ARN:

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

Para especificar todas as chaves que pertencem a uma conta específica, use o caractere curinga (*):

```
"Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/*"
```

Algumas ações AWS de criptografia de pagamento, como as de criação de chaves, não podem ser executadas em um recurso específico. Nesses casos, é necessário utilizar o caractere curinga (*).

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

Para especificar vários recursos em uma única instrução, use uma vírgula como mostrado abaixo:

```
"Resource": [
      "resource1",
      "resource2"
```

Exemplos

Para ver exemplos de políticas baseadas em identidade de criptografia de AWS pagamento, consulte. AWS Exemplos de políticas baseadas em identidade de criptografia de pagamento

Autorização baseada em AWS Tags de criptografia de pagamento

Você pode anexar tags aos recursos AWS de criptografia de pagamento ou passar tags em uma solicitação para criptografia AWS de pagamento. Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `payment-cryptography:ResourceTag/key-name`, `aws:RequestTag/key-name` ou chaves de condição `aws:TagKeys`.

AWS Exemplos de políticas baseadas em identidade de criptografia de pagamento

Por padrão, os usuários e funções do IAM não têm permissão para criar ou modificar recursos AWS de criptografia de pagamento. Eles também não podem realizar tarefas usando a AWS API Console de gerenciamento da AWS AWS CLI, ou. Um administrador do IAM deve criar políticas do IAM que concedam aos usuários e perfis permissão para executarem operações de API específicas nos recursos especificados de que precisam. O administrador deve anexar essas políticas aos usuários ou grupos do IAM que exigem essas permissões.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documentos de política JSON, consulte Criar políticas na guia JSON no Guia do usuário do IAM.

Tópicos

- Práticas recomendadas de política
- Usar o AWS Console de criptografia de pagamento
- Permitir que os usuários visualizem suas próprias permissões
- Capacidade de acessar todos os aspectos do AWS Payment Cryptography
- Capacidade de chamar APIs usando chaves especificadas
- Capacidade de negar um recurso específico

Práticas recomendadas de política

Identity-based as políticas determinam se alguém pode criar, acessar ou excluir recursos AWS de criptografia de pagamento em sua conta. Essas ações podem incorrer em custos para sua Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para saber mais, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do usuário do IAM.
- Aplique permissões de privilégio mínimo: ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para saber mais sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso: é possível adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, é possível escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como CloudFormation. Para saber mais, consulte Elementos da política JSON do IAM: condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas

sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para saber mais, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.

- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para saber mais, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para saber mais sobre as práticas recomendadas do IAM, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Usar o AWS Console de criptografia de pagamento

Para acessar o console AWS de criptografia de pagamento, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre os recursos AWS de criptografia de pagamento em sua AWS conta. Se você criar uma política baseada em identidade que seja mais restritiva que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis do IAM) com essa política.

Para garantir que essas entidades ainda possam usar o console AWS de criptografia de pagamento, anexe também a seguinte política AWS gerenciada às entidades. Para obter mais informações, consulte Adicionar permissões a um usuário no Guia do usuário do IAM.

Você não precisa permitir permissões mínimas do console para usuários que estão fazendo chamadas somente para a API AWS CLI ou para a AWS API. Em vez disso, permita o acesso somente às ações que correspondem à operação da API que você está tentando executar.

Permitir que os usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou 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": "*"
        }
    ]
}
```

Capacidade de acessar todos os aspectos do AWS Payment Cryptography

> **Warning**
>
> Este exemplo fornece permissões amplas e não é recomendado. Em vez disso, considere modelos de acesso menos privilegiados.

Neste exemplo, você quer conceder a um usuário do IAM em sua AWS conta acesso a todas as suas chaves de criptografia de AWS pagamento e a capacidade de chamar todas as APIs de criptografia de AWS pagamento, incluindo ambas `ControlPlane` e operações. `DataPlane`

JSON

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

Capacidade de chamar APIs usando chaves especificadas

Neste exemplo, você deseja conceder a um usuário do IAM em sua AWS conta acesso a uma de suas chaves de criptografia de AWS pagamento `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` e, em seguida, usar esse recurso em duas APIs, e. `GenerateCardValidationData` `VerifyCardValidationData` Por outro lado, o usuário do IAM não terá acesso para usar essa chave em outras operações, como `DeleteKey` ou `ExportKey`

Os recursos podem ser chaves, prefixadas com `key` ou aliases, prefixados com `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"
                ]
            }
        ]
    }
```

Capacidade de negar um recurso específico

> **Warning**
>
> Considere cuidadosamente as implicações da concessão de acesso com caracteres curinga. Em vez disso, considere um modelo de privilégio mínimo.

Neste exemplo, você quer permitir que um usuário do IAM em sua AWS conta acesse qualquer chave de criptografia de AWS pagamento, mas quer negar permissões para uma chave específica. O usuário terá acesso a `VerifyCardData` e `GenerateCardData` com todas as chaves, com exceção da especificada na declaração de negação.

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 políticas para AWS Payment Cryptography

Resource-based políticas são documentos de política JSON que você anexa a um recurso, como uma chave de criptografia AWS de pagamento. Em uma política baseada em recursos, você especifica quem pode acessar a chave e as ações que eles podem executar nela. É possível usar políticas baseadas em recursos para:

- Conceda acesso a uma única chave para vários usuários e funções.
- Conceda acesso a usuários ou funções em outras AWS contas.

Tópicos

- Considerações
- Gerenciando políticas baseadas em recursos
- Resource-based exemplos de políticas

Quando você anexa uma política baseada em recursos a uma chave de criptografia de AWS pagamento, a criptografia de AWS pagamento usa a lógica de avaliação da política do IAM para determinar se um determinado principal está autorizado a realizar a ação solicitada. Para habilitar o acesso entre contas, você pode especificar uma conta inteira ou entidades do IAM em outra conta como principal em uma política baseada em recursos. Cross-account o acesso requer duas políticas:

1. Resource-based política (conta do proprietário da chave) — O proprietário da chave usa `PutResourcePolicy` para conceder acesso à conta do chamador ou ao diretor do IAM.

2. Identity-based política (conta do chamador) — O administrador do IAM do chamador também deve permitir a ação de criptografia de AWS pagamento (por exemplo,`payment-cryptography:EncryptData`) na política do IAM do chamador.

Ambas as políticas devem permitir a ação. Se alguma delas estiver ausente, a solicitação entre contas será negada com`AccessDeniedException`.

Se uma política baseada em recursos conceder acesso a um principal na mesma conta, nenhuma política adicional baseada em identidade será necessária. Para obter mais informações, consulte Como as funções do IAM diferem das Resource-based políticas no Guia do usuário do IAM.

⚠ Operações do plano de controle de políticas de recursos

Resource-based as políticas não se aplicam às operações do plano de controle de políticas de recursos `PutResourcePolicy``GetResourcePolicy`, como, `DeleteResourcePolicy`e. Isso evita possíveis cenários de bloqueio em que uma política de recursos possa negar a capacidade de modificar ou remover a própria política. O acesso a essas operações do plano de controle é regido exclusivamente pelas políticas baseadas em identidade do IAM.

Considerações

Lembre-se do seguinte ao usar políticas baseadas em recursos com criptografia de AWS pagamento.

- AWS A criptografia de pagamento não impõe automaticamente o acesso público às chaves. Você não pode criar uma política baseada em recursos que conceda acesso a diretores anônimos ou públicos. Todo acesso às chaves AWS de criptografia de pagamento requer AWS diretores autenticados, e o acesso público está sempre bloqueado.
- Resource-based as políticas são aplicadas por chave. Cada chave AWS de criptografia de pagamento pode ter no máximo uma política baseada em recursos anexada a ela.
- Resource-based as políticas não se aplicam aos aliases. Quando você faz referência a uma chave por seu alias, a política de recursos anexada à chave subjacente é avaliada.
- Resource-based as políticas não se aplicam às chaves de região de réplica somente para leitura criadas usando Multi-Region a replicação de chaves no momento. As políticas de recursos só podem ser anexadas à chave da Região Primária.

- O `Resource` elemento em uma política baseada em recursos deve ser "*" ou corresponder exatamente ao ARN da chave à qual a política está anexada. "*"O uso é recomendado porque permite que o mesmo documento de política seja reutilizado em várias chaves.
- As APIs de gerenciamento de políticas de recursos (`PutResourcePolicy``GetResourcePolicy`,, e`DeleteResourcePolicy`) são restritas ao Conta da AWS proprietário da chave. Somente os diretores da conta do proprietário da chave podem gerenciar as políticas de recursos.

Gerenciando políticas baseadas em recursos

Você pode gerenciar políticas baseadas em recursos para chaves de criptografia AWS de pagamento usando a AWS CLI API ou. AWS Para usar esse comando, substitua o comando *`italicized placeholder text`* no exemplo por suas próprias informações.

Anexe uma política baseada em recursos

Use a ação da `PutResourcePolicy`API ou o comando da put-resource-policyCLI para anexar uma política baseada em recursos a uma chave. Se uma política já existir, o comando a substituirá.

O exemplo a seguir anexa uma política baseada em recursos de um arquivo JSON a uma chave.

```
aws payment-cryptography put-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --policy file://policy.json
```

Recupere uma política baseada em recursos

Use a ação da `GetResourcePolicy`API ou o comando da get-resource-policyCLI para recuperar a política baseada em recursos anexada a uma chave.

O exemplo a seguir recupera a política baseada em recursos anexada a uma chave.

```
aws payment-cryptography get-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

A resposta retorna o documento de política:

```
{
```

```
    "Policy": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
                },
                "Action": [
                    "payment-cryptography:EncryptData",
                    "payment-cryptography:DecryptData"
                ],
                "Resource": "*"
            }
        ]
    }
}
```

Excluir uma política baseada em recursos

Use a ação da `DeleteResourcePolicy`API ou o comando da delete-resource-policyCLI para remover a política baseada em recursos de uma chave.

O exemplo a seguir exclui a política baseada em recursos anexada a uma chave.

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

Resource-based exemplos de políticas

Conceder acesso entre contas a uma chave

A política baseada em recursos a seguir concede a uma função em outra AWS conta permissão para usar uma chave de criptografia AWS de pagamento para operações criptográficas.

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

```
                "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
            },
            "Action": [
                "payment-cryptography:GenerateCardValidationData",
                "payment-cryptography:VerifyCardValidationData"
            ],
            "Resource": "*"
        }
    ]
}
```

Conceda permissões diferentes para contas diferentes

A política baseada em recursos a seguir demonstra como conceder permissões diferentes aos diretores em contas separadas. Neste exemplo, um 3DS Access Control Server (ACS) em uma conta pode gerar dados de validação de cartão, enquanto um serviço de autorização de pagamento em uma conta diferente só pode validar criptogramas 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 aprovação para criptografia AWS de pagamento

AWS A criptografia de pagamento se integra à Multi-party aprovação (MPA), um recurso da Amazon Web Services Organizations, para ajudar a proteger operações críticas por meio de um processo de aprovação distribuído. Com o MPA, você pode exigir que várias pessoas confiáveis aprovem operações específicas de criptografia AWS de pagamento antes que elas sejam realizadas.

Tópicos

- Visão geral do
- Operações protegidas
- Pré-requisitos
- Ativando e desativando o MPA
- Introdução
- Exemplo: importar um certificado raiz com o MPA ativado
- AWS CloudTrail registro de eventos MPA
- Verificando o status da solicitação e lidando com falhas

Visão geral do

Multi-party a aprovação adiciona uma camada adicional de segurança às operações confidenciais de criptografia de AWS pagamento, exigindo a aprovação de um grupo de pessoas confiáveis antes que a operação possa prosseguir. Isso ajuda a proteger contra alterações não autorizadas se um único conjunto de credenciais for comprometido e impede que um único indivíduo faça uma alteração unilateralmente.

Uma equipe de aprovação é um grupo de aprovadores em sua organização que você designa para aprovar ou negar solicitações de operação protegidas. O processo de aprovação é totalmente gerenciado pelos aprovadores da sua organização. Nenhum AWS funcionário está envolvido na aprovação ou negação de solicitações.

Quando o MPA está habilitado para uma operação protegida, ocorre o seguinte:

1. Um solicitante inicia a operação protegida.

2. O MPA cria uma sessão de aprovação e notifica os membros da equipe de aprovação.
3. Os membros da equipe de aprovação analisam a solicitação e a aprovam ou negam por meio do portal MPA.
4. Quando o limite mínimo exigido de aprovações é atingido, a operação prossegue. Se a solicitação for negada pela equipe de aprovação ou o tempo permitido da sessão expirar antes que o limite de aprovação seja atingido, a operação não será executada. Em ambos os casos, o solicitante deve enviar uma nova solicitação para repetir a operação.

> **Note**
>
> Ao importar um certificado CA raiz com o MPA habilitado, o `RequesterComment` parâmetro é obrigatório. Esse comentário está incluído na notificação de aprovação enviada à equipe de aprovação, fornecendo contexto para a solicitação.

Operações protegidas

AWS A criptografia de pagamento suporta MPA para a seguinte operação:

- `ImportKey`com material de `RootCertificatePublicKey` chave — Importar um certificado de chave pública raiz é uma operação crítica porque os certificados raiz estabelecem a âncora confiável para todas as importações e exportações de chaves subsequentes usando a troca de chaves assimétrica, como. TR-34 Exigir a aprovação de várias partes para essa operação ajuda a garantir que nenhum indivíduo possa estabelecer ou alterar unilateralmente a raiz da confiança em suas chaves de criptografia de AWS pagamento.

Pré-requisitos

Antes de usar o MPA com criptografia AWS de pagamento, você deve preencher os seguintes pré-requisitos:

- Configure o MPA em seu ambiente Amazon Web Services Organizations. Para obter instruções, consulte O que é Multi-party aprovação? no Guia do usuário de Multi-party aprovação.
- Crie pelo menos uma equipe de aprovação com os aprovadores necessários.
- Compartilhe a equipe de aprovação com a Conta da AWS que contém suas chaves AWS de criptografia de pagamento usando AWS Resource Access Manager.

- A conta de gerenciamento em sua organização deve ser Multi-party aprovada.

Ativando e desativando o MPA

Depois de configurar uma equipe de aprovação, você pode ativar o MPA para criptografia AWS de pagamento associando a equipe à sua conta. Você também pode desativar o MPA desassociando a equipe, embora a dissociação exija a aprovação da equipe de aprovação atualmente associada.

Ativar MPA

Use a ação da `AssociateMpaTeam` API ou o comando da associate-mpa-team CLI para associar uma equipe de aprovação à sua conta de criptografia AWS de pagamento. Uma vez associadas, as operações protegidas exigem a aprovação da equipe antes de poderem prosseguir.

```
aws payment-cryptography associate-mpa-team \
    --team-arn arn:aws:mpa:us-east-1:111122223333:team/my-approval-team
```

Desativar MPA

Use a ação `DisassociateMpaTeam` da API ou o comando da disassociate-mpa-team CLI para remover a associação da equipe de aprovação. A dissociação de uma equipe é, em si, uma operação protegida que requer aprovação da equipe de aprovação atualmente associada.

```
aws payment-cryptography disassociate-mpa-team \
    --team-arn arn:aws:mpa:us-east-1:111122223333:team/my-approval-team
```

> **Important**
>
> A desativação do MPA exige a aprovação da equipe de aprovação atualmente associada. Isso garante que nenhum indivíduo possa remover unilateralmente a proteção de aprovação multipartidária.

> **Note**
>
> O `--requester-comment` parâmetro é opcional para associate-mpa-team disassociate-mpa-team e.

Introdução

Para começar a usar o MPA for AWS Payment Cryptography, consulte o Guia do usuário de Multi-party aprovação para obter instruções detalhadas de configuração, incluindo como criar equipes de aprovação, configurar políticas de aprovação e gerenciar sessões de aprovação.

Exemplo: importar um certificado raiz com o MPA ativado

Depois que o MPA for ativado e uma equipe de aprovação for associada à `ImportKey` operação`RootCertificatePublicKey`, a solicitação de importação precisará de aprovação antes de prosseguir.

1. Um solicitante liga `import-key` para importar um certificado de chave pública raiz. Para usar esse comando, substitua o comando *`italicized placeholder text`* no exemplo por suas próprias informações.

```
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"
```

   A resposta retorna uma chave `KeyState` definida como`CREATE_IN_PROGRESS`, indicando que a solicitação está aguardando aprovação. A resposta também inclui `MpaStatus` detalhes sobre a sessão de aprovação:

```
{
    "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. Como o MPA está habilitado, a solicitação não é concluída imediatamente. Em vez disso, a criptografia de AWS pagamento cria uma sessão de aprovação e retorna uma resposta indicando que a aprovação está pendente.
3. Os membros da equipe de aprovação recebem uma notificação e analisam a solicitação por meio do portal MPA. Depois que o número necessário de aprovadores aprova a solicitação, a operação de importação prossegue e o certificado raiz é importado.

AWS CloudTrail registro de eventos MPA

Quando o MPA está ativado, a criptografia AWS de pagamento registra os eventos do serviço AWS CloudTrailquando uma sessão de aprovação é concluída. Esses eventos registram o resultado do processo de aprovação, incluindo se a solicitação foi aprovada ou falhou. Você pode usar esses registros para auditar a atividade do MPA e rastrear o status das operações protegidas.

MPA-related CloudTrail os eventos incluem os seguintes campos em`serviceEventDetails`:

- `keyArn`— O ARN da chave afetada pela operação.
- `operation`— A operação protegida que foi solicitada.
- `mpaSessionArn`— O ARN da sessão de aprovação do MPA.
- `sessionStatus`— O resultado da sessão de aprovação (`APPROVED`ou`FAILED`).

Solicitação aprovada

O exemplo a seguir mostra um CloudTrail evento para uma `ImportKey` solicitação que foi aprovada pela equipe do 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"
    }
}
```


Solicitação falhada

O exemplo a seguir mostra um CloudTrail evento para uma `ImportKey` solicitação que foi negada ou atingiu o tempo limite:


```
{
    "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"
    }
}
```

Para saber mais AWS CloudTrail, consulte o Guia AWS CloudTrail do usuário.

Verificando o status da solicitação e lidando com falhas

Você pode verificar o status de uma solicitação de MPA pendente ligando para `GetKey`. A resposta inclui o `MpaStatus` campo com os detalhes da sessão de aprovação atual. Para usar esse comando, substitua o comando *`italicized placeholder text`* no exemplo por suas próprias informações.

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

Enquanto a solicitação estiver pendente de aprovação, a resposta será exibida `KeyState` como `CREATE_IN_PROGRESS` e `MpaStatus.Status` como`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"
        }
    }
}
```

Depois que o número necessário de aprovadores aprova a solicitação, ela `KeyState` passa para `CREATE_COMPLETE` e `MpaStatus.Status` para. `APPROVED` A chave agora está pronta para uso:

```
{
    "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"
        }
    }
}
```

Se a solicitação for negada pela equipe de aprovação ou a sessão expirar antes que o limite de aprovação seja atingido, as `KeyState` alterações serão alteradas `CREATE_FAILED` e `MpaStatus.Status` alteradas em: `FAILED`

```
{
    "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"
        }
    }
}
```

Uma chave no `CREATE_FAILED` status não pode ser usada para operações criptográficas. Para repetir a importação, você deve enviar uma nova `ImportKey` solicitação, que criará uma nova sessão de aprovação.

Solução de problemas AWS Criptografia de pagamento, identidade e acesso

Tópicos serão adicionados a esta seção à medida que IAM-related problemas específicos da criptografia AWS de pagamento forem identificados. Para obter conteúdo geral de solução de problemas sobre tópicos do IAM, consulte a seção de solução de problemas do Guia do usuário do IAM.

Monitorando a criptografia AWS de pagamento

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho da criptografia de AWS pagamento e de suas outras soluções da AWS. A AWS fornece as seguintes ferramentas de monitoramento para monitorar a criptografia de AWS pagamentos, relatar quando algo está errado e realizar ações automáticas quando apropriado:

- A Amazon CloudWatch monitora seus AWS recursos e os aplicativos em que você executa AWS em tempo real. Você pode coletar e rastrear métricas, criar painéis personalizados e definir alarmes que o notificam ou que realizam ações quando uma métrica especificada atinge um limite definido. Por exemplo, você pode CloudWatch rastrear o uso de determinados APIs ou notificá-lo se estiver se aproximando de suas cotas de criptografia AWS de pagamento. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.
- O Amazon CloudWatch Logs permite que você monitore, armazene e acesse seus arquivos de log de EC2 instâncias da Amazon e de outras fontes. CloudTrail CloudWatch Os registros podem monitorar as informações nos arquivos de log e notificá-lo quando determinados limites forem atingidos. É possível também arquivar seus dados de log em armazenamento resiliente. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch Logs.
- AWS CloudTrailcaptura chamadas de API e eventos relacionados feitos por ou em nome de sua AWS conta e entrega os arquivos de log para um bucket do Amazon S3 que você especificar. Você pode identificar quais usuários e contas ligaram AWS, o endpoint chamado, os recursos (chaves) usados, o endereço IP de origem a partir do qual as chamadas foram feitas e quando as chamadas ocorreram. Para obter mais informações, consulte o Guia do usuário do AWS CloudTrail.

Tópicos

- Registrando chamadas da API de criptografia de AWS pagamento usando AWS CloudTrail

Registrando chamadas da API de criptografia de AWS pagamento usando AWS CloudTrail

AWS A criptografia de pagamento é integrada com AWS CloudTrail um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço na criptografia de AWS pagamento. CloudTrail captura todas as chamadas de API para criptografia AWS de pagamento

como eventos. As chamadas capturadas incluem as aquelas do console e chamadas de código para operações API da . Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para criptografia de AWS pagamento. Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes de gerenciamento (Plano de Controle) no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita à AWS Payment Cryptography, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o Guia AWS CloudTrail do usuário.

Tópicos

- AWS Informações de criptografia de pagamento em CloudTrail
- Controle os eventos do avião em CloudTrail
- Eventos de dados em CloudTrail
- Compreendendo as entradas do arquivo de log do AWS Payment Cryptography Control Plane
- Compreendendo as entradas do arquivo de log do plano de dados de criptografia de AWS pagamento

AWS Informações de criptografia de pagamento em CloudTrail

CloudTrail é ativado em sua AWS conta quando você cria a conta. Quando a atividade ocorre na criptografia de AWS pagamento, essa atividade é registrada em um CloudTrail evento junto com outros eventos de AWS serviço no histórico de eventos. É possível visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para um registro contínuo dos eventos em sua AWS conta, incluindo eventos para criptografia AWS de pagamento, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, quando uma trilha é criada no console, a mesma é aplicada a todas as regiões da AWS . A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para saber mais, consulte:

- Visão geral da criação de uma trilha

- CloudTrail serviços e integrações suportados
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões
- Recebendo arquivos de CloudTrail log de várias contas

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar o seguinte:

- Se a solicitação foi feita com credenciais de usuário root ou AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail .

Controle os eventos do avião em CloudTrail

CloudTrail registra operações de criptografia de AWS pagamento, como CreateKey,, ImportKey, DeleteKeyListKeysTagResource, e todas as outras operações do plano de controle.

Eventos de dados em CloudTrail

Os eventos de dados fornecem informações sobre as operações de recursos realizadas em ou em um recurso, como criptografar uma carga ou traduzir um pino. Eventos de dados são atividades de alto volume que CloudTrail não são registradas por padrão. Você pode ativar o registro de ações da API de eventos de dados para eventos do plano de dados de criptografia de AWS pagamento usando CloudTrail APIs nosso console. Para obter mais informações, consulte Registrar eventos de dados, no Guia do usuário do AWS CloudTrail .

Com CloudTrail, você deve usar seletores de eventos avançados para decidir quais atividades da API AWS de criptografia de pagamento são registradas e registradas. Para registrar eventos do plano de dados de criptografia de AWS pagamento, você deve incluir o tipo de recurso `AWS Payment Cryptography key` e. `AWS Payment Cryptography alias` Depois de configurado, é possível refinar ainda mais suas preferências de registro em log selecionando eventos de dados específicos para gravação, como usar o filtro `eventName` para rastrear eventos `EncryptData`.

Para obter mais informações, consulte `AdvancedEventSelector` na Referência de APIs do AWS CloudTrail .

> **Note**
>
> Para se inscrever em eventos de dados de criptografia de AWS pagamento, você deve utilizar seletores de eventos avançados. Recomendamos se inscrever em eventos importantes e de alias para garantir que você receba todos os eventos.

AWS Eventos de dados de criptografia de pagamento:

- `DecryptData`
- `EncryptData`
- `GenerateCardValidationData`
- `GenerateMac`
- `GeneratePinData`
- `ReEncryptData`
- `TranslatePinData`
- `VerifyAuthRequestCryptogram`
- `VerifyCardValidationData`
- `VerifyMac`
- `VerifyPinData`

Há cobranças adicionais para eventos de dados. Para obter mais informações, consulte AWS CloudTrail Preço.

Compreendendo as entradas do arquivo de log do AWS Payment Cryptography Control Plane

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante.

CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a ação Criptografia AWS `CreateKey` de pagamento.

```
      {
    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={
    }
  }
}
```

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a criptografia de AWS pagamento que permite a replicação de chaves em várias regiões.

```
{
    "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"
}
```

Compreendendo as entradas do arquivo de log do plano de dados de criptografia de AWS pagamento

Os eventos do plano de dados podem opcionalmente ser configurados e funcionar de forma semelhante aos registros do plano de controle, mas normalmente são volumes muito maiores. Dada a natureza confidencial de algumas entradas e saídas das operações do plano de dados de criptografia de AWS pagamento, você pode encontrar determinados campos com a mensagem "*** Dados confidenciais editados ***". Isso não é configurável e tem como objetivo evitar que dados confidenciais apareçam em registros ou trilhas.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a ação Criptografia AWS `EncryptData` de pagamento.

```
      {
        "Records": [
            {
                "eventVersion": "1.09",
                "userIdentity": {
                    "type": "AssumedRole",
                    "principalId": "TESTXECZ5U2ZULLHHMJG:DataPlane-User",
                    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/DataPlane-
User",
```

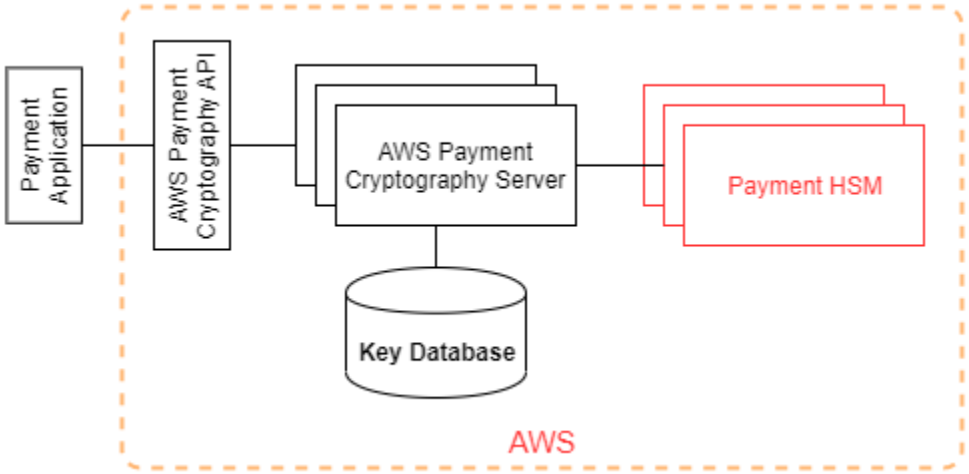
```
                    "accountId": "111122223333",
                    "accessKeyId": "TESTXECZ5U2ZULLHHMJG",
                    "userName": "",
                    "sessionContext": {
                        "sessionIssuer": {
                            "type": "Role",
                            "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"
                }
            }
        ]
    }
```

Detalhes criptográficos

AWS A criptografia de pagamento fornece uma interface web para gerar e gerenciar chaves criptográficas para transações de pagamento. AWS A criptografia de pagamento oferece serviços padrão de gerenciamento de chaves e criptografia de transações de pagamento e ferramentas que você pode usar para gerenciamento e auditoria centralizados. Esta documentação fornece uma descrição detalhada das operações criptográficas que você pode usar na Criptografia de AWS Pagamento para ajudá-lo a avaliar os recursos oferecidos pelo serviço.

AWS A criptografia de pagamento contém várias interfaces (incluindo uma API RESTful, por meio da AWS CLI, do AWS SDK e Console de gerenciamento da AWS do) para solicitar operações criptográficas de uma frota distribuída de módulos de segurança de hardware PCI PTS. HSM-validated



AWS A criptografia de pagamento é um serviço hierárquico que consiste em hosts de criptografia de AWS pagamento voltados para a web e um nível de HSMs. O agrupamento desses hosts em camadas forma a pilha de criptografia de AWS pagamento. Todas as solicitações de criptografia AWS de pagamento devem ser feitas pelo protocolo Transport Layer Security (TLS) e encerradas em um host de criptografia de AWS pagamento. Os hosts de serviços permitem TLS apenas com um pacote de criptografia que fornece perfect forward secrecy. O serviço autentica e autoriza suas solicitações usando os mesmos mecanismos de credenciais e políticas do IAM que estão disponíveis para todas as outras AWS operações de API.

AWS Os servidores de criptografia de pagamento se conectam ao HSM subjacente por meio de uma rede privada e não virtual. As conexões entre os componentes do serviço e o HSM são protegidas com TLS mútuo (mTLS) para autenticação e criptografia.

Tópicos

- Objetivos de projeto
- Fundamentos
- Operações internas
- Operações do cliente

Objetivos de projeto

AWS A criptografia de pagamento foi projetada para atender aos seguintes requisitos:

- Confiável: o uso de chaves é protegido por políticas de controle de acesso que você define e gerencia. Não há mecanismo para exportar chaves de criptografia de AWS pagamento em texto simples. A confidencialidade das suas chaves criptográficas é crucial. Vários funcionários da Amazon com acesso específico por função aos controles de acesso baseados em quórum são obrigados a realizar ações administrativas no. HSMs Nenhum funcionário da Amazon tem acesso às chaves principais (ou mestras) ou backups do HSM. As chaves principais não podem ser sincronizadas com as HSMs que não fazem parte de uma região de criptografia de AWS pagamento. Todas as outras chaves são protegidas pelas chaves principais do HSM. Portanto, as chaves AWS de criptografia de pagamento do cliente não podem ser usadas fora do serviço de criptografia de AWS pagamento que opera na conta do cliente.
- Baixa latência e alta taxa de transferência — A criptografia de AWS pagamento fornece operações criptográficas em nível de latência e taxa de transferência adequadas para gerenciar chaves criptográficas de pagamento e processar transações de pagamento.
- Durabilidade: a durabilidade das chaves criptográficas foi projetada para ser igual à dos serviços de maior durabilidade na AWS. Uma única chave criptográfica pode ser compartilhada com um terminal de pagamento, cartão com chip EMV ou outro dispositivo criptográfico seguro (SCD) que esteja em uso por muitos anos.
- Regiões independentes: a AWS fornece regiões independentes a clientes que precisam restringir o acesso a dados em diferentes regiões ou precisam atender a requisitos de residência de dados. O uso de chaves pode ser isolado dentro de uma região da AWS.
- Fonte segura de números aleatórios — Como a criptografia forte depende da geração de números aleatórios verdadeiramente imprevisível, a criptografia de AWS pagamento fornece uma fonte validada e de alta qualidade de números aleatórios. Toda a geração de chaves para criptografia AWS de pagamento usa HSM listado no PCI PTS HSM, operando no modo PCI.

- Auditoria — A criptografia de AWS pagamento registra o uso e o gerenciamento de chaves criptográficas em CloudTrail registros e registros de serviços disponíveis na Amazon. CloudWatch Você pode usar CloudTrail registros para inspecionar o uso de suas chaves criptográficas, incluindo o uso de chaves por contas com as quais você compartilhou chaves. AWS A criptografia de pagamento é auditada por avaliadores terceirizados de acordo com os padrões aplicáveis de segurança de pagamento PCI, marca de cartão e padrões regionais de segurança de pagamento. Atestados e guias de responsabilidade compartilhada estão disponíveis no AWS Artifact.
- Elastic — A criptografia de AWS pagamento se expande e aumenta de acordo com sua demanda. Em vez de prever e reservar a capacidade do HSM, a criptografia de pagamento fornece criptografia AWS de pagamento sob demanda. AWS A criptografia de pagamento assume a responsabilidade de manter a segurança e a conformidade do HSM para fornecer capacidade suficiente para atender aos picos de demanda do cliente.

Fundamentos

Os tópicos deste capítulo descrevem as primitivas criptográficas da criptografia de AWS pagamento e onde elas são usadas. Eles também apresentam os elementos básicos do serviço.

Tópicos

- Primitivas criptográficas
- Entropia e geração de números aleatórios
- Operações de chave simétrica
- Operações de chave assimétrica
- Armazenamento de chaves
- Importar chaves usando chaves simétricas
- Importar chaves usando chaves assimétricas
- Exportação de chaves
- Protocolo de chave única derivada por transação (DUKPT)
- Hierarquia de chaves

Primitivas criptográficas

AWS A criptografia de pagamento usa algoritmos criptográficos padrão e parametrizáveis para que os aplicativos possam implementar os algoritmos necessários para seu caso de uso. O conjunto de

algoritmos criptográficos é definido pelos padrões PCI, ANSI X9, EMVCo e ISO. Toda a criptografia é executada por HSMs listados no padrão PCI PTS HSM em execução no modo PCI.

Entropia e geração de números aleatórios

AWS A geração da chave de criptografia de pagamento é realizada nos HSMs AWS de criptografia de pagamento. Os HSMs implementam um gerador de números aleatórios que atende aos requisitos do PCI PTS HSM para todos os tipos e parâmetros de chave compatíveis.

Operações de chave simétrica

Algoritmos de chave simétrica e pontos fortes de chave definidos em ANSI X9 TR 31, ANSI e PCI X9.24 PIN Anexo C são suportados:

- Funções de hash: algoritmos da família SHA2 e SHA3 com tamanho de saída maior que 2551. Exceto pela compatibilidade retroativa com terminais pré-PCI PTS POI v3.
- Criptografia e descriptografia: AES com tamanho de chave maior ou igual a 128 bits ou TDEA com tamanho de chave maior ou igual a 112 bits (2 chaves ou 3 chaves).
- Códigos de autenticação de mensagens (MACs) CMAC ou GMAC com AES, bem como HMAC com uma função hash aprovada e um tamanho de chave maior ou igual a 128.

AWS A criptografia de pagamento usa AES 256 para chaves principais do HSM, chaves de proteção de dados e chaves de sessão TLS.

Nota: Algumas das funções listadas são usadas internamente para oferecer suporte a protocolos e estruturas de dados padrão. Consulte a documentação da API para ver os algoritmos suportados por ações específicas.

Operações de chave assimétrica

Algoritmos de chave assimétrica e pontos fortes de chave definidos em ANSI X9 TR 31, ANSI e PCI PIN X9.24 Anexo C são suportados:

- Esquemas de estabelecimento de chaves aprovados — conforme descrito em NIST SP800-56A (ECC/FCC2-based acordo-chave), NIST SP800-56B (IFC-based acordo-chave) e NIST SP800-38F (AES-based chave). encryption/wrapping

AWS Os hosts de criptografia de pagamento só permitem conexões com o serviço usando TLS com um pacote de criptografia que fornece sigilo direto perfeito.

Nota: Algumas das funções listadas são usadas internamente para oferecer suporte a protocolos e estruturas de dados padrão. Consulte a documentação da API para ver os algoritmos suportados por ações específicas.

Armazenamento de chaves

AWS As chaves de criptografia de pagamento são protegidas pelas chaves principais HSM AES 256 e armazenadas em blocos de chaves ANSI X9 TR 31 em um banco de dados criptografado. O banco de dados é replicado para um banco de dados na memória em servidores de criptografia AWS de pagamento.

De acordo com o Anexo C do Regulamento de segurança PCI PIN, as chaves AES 256 são tão ou mais fortes que:

- TDEA de 3 chaves
- RSA de 15360 bits
- ECC de 512 bits
- DSA, DH e MQV 15360/512

Importar chaves usando chaves simétricas

AWS A criptografia de pagamento suporta a importação de criptogramas e blocos de chaves com chaves simétricas ou públicas com uma chave de criptografia de chave simétrica (KEK) tão forte ou mais forte que a chave protegida para importação.

Importar chaves usando chaves assimétricas

AWS A criptografia de pagamento suporta a importação de criptogramas e blocos de chaves com chaves simétricas ou públicas protegidas por uma chave de criptografia de chave privada (KEK) tão forte ou mais forte que a chave protegida para importação. A chave pública fornecida para a decodificação deve ter sua autenticidade e integridade garantidas por um certificado de uma autoridade em que o cliente confie.

Os KEK públicos fornecidos pela AWS Payment Cryptography têm a autenticação e a proteção de integridade de uma autoridade de certificação (CA) com conformidade atestada com o PCI PIN Security e o PCI P2PE Anexo A.

Exportação de chaves

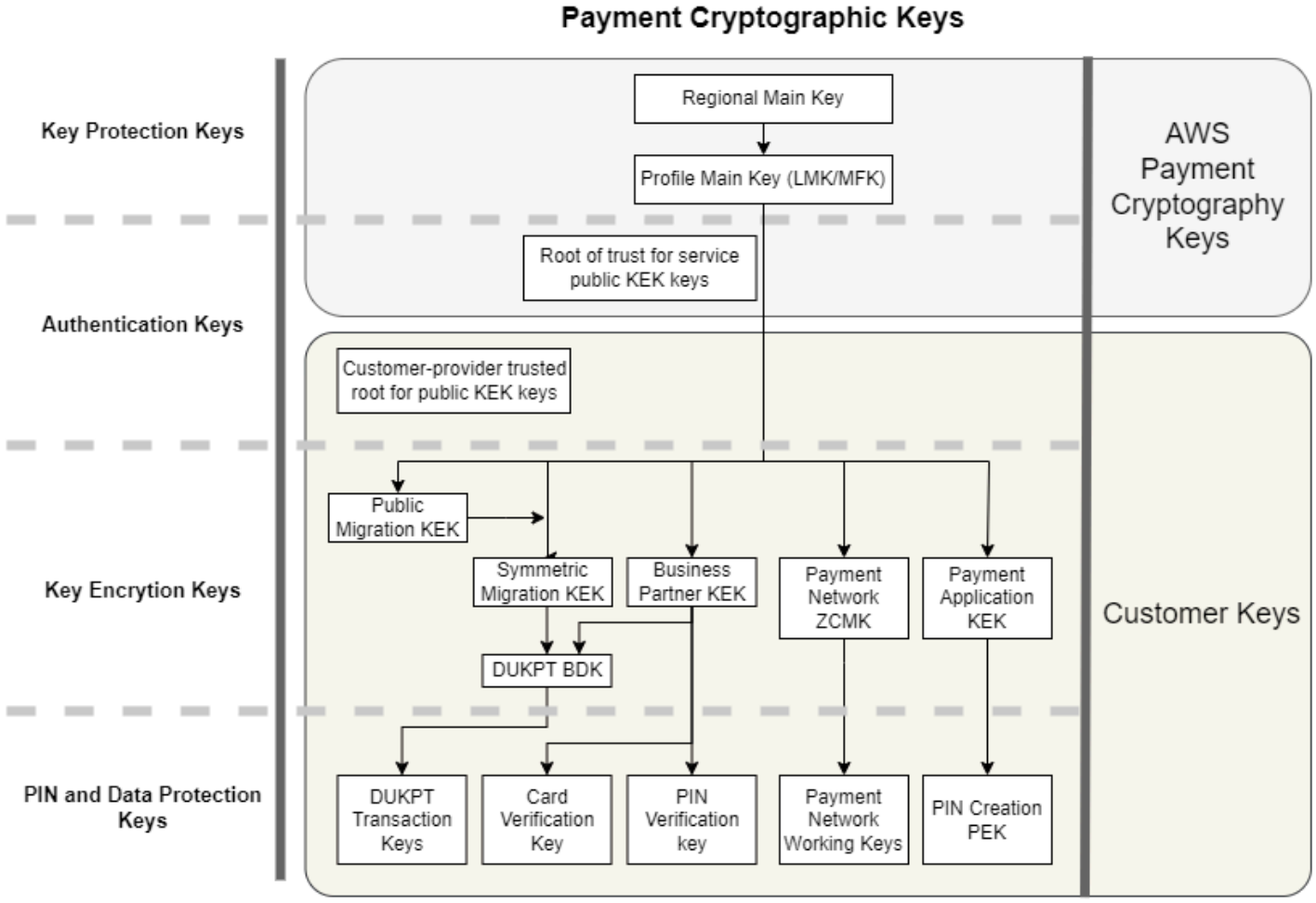
As chaves podem ser exportadas e protegidas por chaves com as chaves apropriadas KeyUsage e que sejam tão fortes ou mais fortes do que a chave a ser exportada.

Protocolo de chave única derivada por transação (DUKPT)

AWS A criptografia de pagamento é compatível com chaves de derivação base (BDK) TDEA e AES, conforme descrito pela ANSI. X9.24-3

Hierarquia de chaves

A hierarquia de chaves de criptografia de AWS pagamento garante que as chaves sejam sempre protegidas por chaves tão fortes ou mais fortes do que as chaves que elas protegem.



AWS As chaves de criptografia de pagamento são usadas para proteção de chaves dentro do serviço:

Chave	Description
Chave principal regional	Protege imagens ou perfis virtuais do HSM usados para processamento criptográfico. Essa chave existe somente no HSM e nos backups seguros.
Chave principal do perfil	Chave de proteção de chave de cliente de alto nível, tradicionalmente chamada de chave mestra local (LMK) ou chave de arquivo mestre (MFK) para chaves de cliente. Essa chave existe somente no HSM e nos backups seguros. Os perfis definem configurações de HSM distintas, conforme exigido pelos padrões de segurança para casos de uso de pagamento s.
Raiz de confiança nas chaves de chave de criptografia de chave pública (KEK) da criptografia de AWS pagamento	A chave pública raiz confiável e o certifica do para autenticar e validar chaves públicas fornecidos pela AWS Payment Cryptogra phy para importação e exportação de chaves usando chaves assimétricas.

As chaves do cliente são agrupadas por chaves usadas para proteger outras chaves e chaves que protegem dados relacionados a pagamentos. Estes são exemplos de chaves de cliente dos dois tipos:

Chave	Description
Customer-provided raiz confiável para chaves KEK públicas	Chave pública e certificado fornecidos por você como raiz de confiança para autenticar e validar chaves públicas que você fornece para importação e exportação de chaves usando chaves assimétricas.
Chaves de criptografia de chaves (KEK)	As KEK são usadas exclusivamente para criptografar outras chaves para troca entre

Chave	Description
	armazenamentos externos de chaves e criptografia de AWS pagamento, parceiros de negócios, redes de pagamento ou aplicativos diferentes em sua organização.
Chave de derivação de base (BDK) da chave única derivada por transação (DUKPT)	As BDKs são usadas para criar chaves exclusivas para cada terminal de pagamento e traduzir transações de vários terminais em uma única chave de trabalho do banco adquirente ou do adquirente. A melhor prática, exigida pela Point-to-Point criptografia PCI (P2PE), é que diferentes BDKs sejam usados para diferentes modelos de terminal, serviços de injeção ou inicialização de chaves ou outra segmentação para limitar o impacto do comprometimento de um BDK.
Chave mestra de controle de zona da rede de pagamento (ZCMK)	As ZCMK, também conhecidas como chaves de zona ou chaves mestras de zona, são fornecidas pelas redes de pagamento para estabelecer as chaves de trabalho iniciais.
Chaves de transação DUKPT	Os terminais de pagamento configurados para DUKPT derivam uma chave única para o terminal e a transação. O HSM que recebe a transação pode determinar a chave a partir do identificador do terminal e do número da sequência da transação.

Chave	Description
Chaves de preparação de dados do cartão	As chaves mestras do emissor EMV, as chaves do cartão EMV e os valores de verificação e as chaves de proteção de arquivos de dados de personalização do cartão são usadas para criar dados para cartões individuais para uso por um provedor de personalização de cartões. Essas chaves e dados de validação criptográfica também são usados pelos bancos emissores ou emissores para autenticar os dados do cartão como parte da autorização de transaçõe s.
Chaves de preparação de dados do cartão	As chaves mestras do emissor EMV, as chaves do cartão EMV e os valores de verificação e as chaves de proteção de arquivos de dados de personalização do cartão são usadas para criar dados para cartões individuais para uso por um provedor de personalização de cartões. Essas chaves e dados de validação criptográfica também são usados pelos bancos emissores ou emissores para autenticar os dados do cartão como parte da autorização de transaçõe s.
Chaves de funcionamento da rede de pagamento	Frequentemente chamadas de chave de trabalho do emissor ou chave de trabalho do adquirente, essas são as chaves que criptogra fam as transações enviadas ou recebidas das redes de pagamento. Essas chaves são alternadas com frequência pela rede, geralmente a cada dia ou de hora em hora. Essas são chaves de criptografia PIN (PEK) para PIN/Debit transações.

Chave	Description
Chaves de criptografia (PEK) do número de identificação pessoal (PIN)	Os aplicativos que criam ou descriptografam blocos de PIN usam PEKs para impedir o armazenamento ou a transmissão de PIN em texto não criptografado.

Operações internas

Este tópico descreve os requisitos internos implementados pelo serviço para proteger as chaves do cliente e as operações criptográficas para um serviço de gerenciamento de chaves e criptografia de pagamento escalável e distribuído globalmente.

Tópicos

- Proteção HSM
- Gerenciamento geral de chaves
- Gerenciamento de chaves de clientes
- Segurança de comunicação
- Registro em log e monitoramento

Proteção HSM

Especificações e ciclo de vida do HSM

AWS A criptografia de pagamento usa uma frota de produtos disponíveis comercialmente. HSMs Eles HSMs são validados pelo FIPS 140-2 Nível 3 e também usam versões de firmware e a política de segurança listadas na lista de dispositivos PCI PTS aprovados pelo PCI Security Standards Council como compatíveis com PCI HSM v3. O padrão PCI PTS HSM inclui requisitos adicionais para fabricação, envio, implantação, gerenciamento e destruição do hardware HSM, que são importantes para a segurança e conformidade dos pagamentos, mas não são abordados pelo FIPS 140.

Avaliadores terceirizados verificam a marca, o modelo, o firmware, a configuração, o gerenciamento físico do ciclo de vida, o controle de alterações, os controles de acesso do operador, o gerenciamento de chaves principais e todos os requisitos de PCI PIN e P2PE relacionados às operações do HSM. HSMs

Todos HSMs são operados no modo PCI e configurados com a política de segurança PCI PTS HSM. Somente as funções necessárias para oferecer suporte aos casos de uso da criptografia de AWS pagamento estão ativadas. AWS A criptografia de pagamento não permite a impressão, exibição ou devolução de texto PINs não criptografado.

Segurança física do dispositivo HSM

Somente HSMs aqueles com chaves de dispositivo assinadas por uma autoridade certificadora de criptografia de AWS pagamento (CA) pelo fabricante antes da entrega podem ser usados pelo serviço. A criptografia AWS de pagamento é uma subCA da CA do fabricante que é a raiz da confiança dos certificados de fabricantes e dispositivos do HSM. A CA do fabricante atestou a conformidade com o Anexo A de Segurança PCI PIN e o Anexo A. O fabricante verifica se todos os HSM com chaves de dispositivo assinadas pela Autoridade de Criptografia de AWS Pagamentos são enviados para o destinatário designado pela AWS.

Conforme exigido pela segurança PCI PIN, o fabricante fornece uma lista de números de série por meio de um canal de comunicação diferente do da remessa do HSM. Esses números de série são verificados em cada etapa do processo de instalação do HSM nos datacenters da AWS. Por fim, os operadores de criptografia de AWS pagamento validam a lista de HSM instalados em relação à lista do fabricante antes de adicionar o número de série à lista de HSM autorizados a receber AWS chaves de criptografia de pagamento.

HSMs estão sempre em armazenamento seguro ou sob controle duplo, o que inclui:

- Remessa do fabricante para uma instalação de montagem de rack da AWS.
- Durante a montagem do rack.
- Envio da instalação de montagem do rack para um datacenter.
- Recebimento e instalação em uma sala de processamento segura do datacenter. Os racks HSM impõem controle duplo com fechaduras controladas por cartão, sensores de porta com alarme e câmeras.
- Durante as operações.
- Durante o descomissionamento e a destruição.

Um completo chain-of-custody, com responsabilidade individual, é mantido e monitorado para cada HSM.

Inicialização do Java

Um HSM só é inicializado como parte da frota de criptografia de AWS pagamento depois que sua identidade e integridade são validadas por números de série, chaves de dispositivo instaladas pelo fabricante e soma de verificação do firmware. Depois que a autenticidade e a integridade de um HSM são validadas, ele é configurado, incluindo a ativação do Modo PCI. Em seguida, as chaves principais da região de criptografia de AWS pagamento e as chaves principais do perfil são estabelecidas e o HSM fica disponível para o serviço.

Serviço e reparo do HSM

O HSM tem componentes que podem ser reparados e não exigem a violação do limite criptográfico do dispositivo. Esses componentes incluem ventiladores de resfriamento, fontes de alimentação e baterias. Se um HSM ou outro dispositivo dentro do rack do HSM precisar de manutenção, o controle duplo será mantido durante todo o período em que o rack estiver aberto.

Descomissionamento do HSM

O descomissionamento ocorre devido end-of-life ou falha de um HSM. Os HSM são logicamente zerados antes de serem removidos do rack e, se estiverem funcionando, são destruídos nas salas de processamento seguras dos datacenters da AWS. Eles nunca são devolvidos ao fabricante para reparo, usados para outra finalidade ou removidos de uma sala de processamento segura antes da destruição.

Atualização de firmware do HSM

As atualizações de firmware do HSM são aplicadas quando necessário para manter o alinhamento com as versões listadas do PCI PTS HSM e do FIPS 140-2 (ou FIPS 140-3), se uma atualização estiver relacionada à segurança ou se for determinado que os clientes podem se beneficiar dos recursos de uma nova versão. AWS A criptografia de pagamento HSMs executa o off-the-shelf firmware, compatível com as versões listadas no PCI PTS HSM. As novas versões de firmware são validadas quanto à integridade com as versões de firmware com certificação PCI ou FIPS e depois testadas quanto à funcionalidade antes da implantação para todos. HSMs

Acesso do operador

Os operadores podem ter acesso sem console ao HSM para solução de problemas em casos raros em que as informações coletadas do HSM durante as operações normais são insuficientes para identificar um problema ou planejar uma mudança. As etapas a seguir são executadas:

- As atividades de solução de problemas são desenvolvidas e aprovadas e a sessão sem console é agendada.
- Um HSM é removido do serviço de processamento do cliente.
- As chaves principais são excluídas, sob controle duplo.
- O operador tem permissão para acessar o HSM sem console para realizar atividades de solução de problemas aprovadas, sob controle duplo.
  - Após o término da sessão sem console, o processo de provisionamento inicial é executado no HSM, retornando o firmware e a configuração padrão e, em seguida, sincronizando a chave principal, antes de devolver o HSM para atender aos clientes.
  - Os registros da sessão são registrados no controle de alterações.
  - As informações obtidas na sessão são usadas para planejar mudanças futuras.

Todos os registros de acesso que não sejam do console são revisados quanto à conformidade do processo e possíveis alterações no monitoramento do HSM, no processo non-console-access de gerenciamento ou no treinamento do operador.

Gerenciamento geral de chaves

Todos os HSMs em uma região são sincronizados com uma chave principal da região. Uma chave principal de região protege pelo menos uma chave principal de perfil. Uma chave principal de perfil protege as chaves do cliente.

Todas as chaves principais são geradas por um HSM e distribuídas por distribuição simétrica de chaves usando técnicas assimétricas, alinhadas com ANSI X9 TR 34 e Anexo A do PCI PIN.

Geração

Chaves principais AES de 256 bits são geradas em um dos HSM provisionados para a frota de HSM de serviço, usando o gerador de números aleatórios PCI PTS HSM.

Sincronização da chave principal da região

As chaves principais da região do HSM são sincronizadas pelo serviço em toda a frota regional com mecanismos definidos pela ANSI X9 TR-34, que incluem:

- Autenticação mútua usando chaves e certificados do host de distribuição de chaves (KDH) e do dispositivo receptor de chaves (KRD) para fornecer autenticação e integridade de chaves públicas.

- Os certificados são assinados por uma autoridade de certificação (CA) que atende aos requisitos do Anexo A2 do PCI PIN, exceto para algoritmos assimétricos e pontos fortes de chave apropriados para proteger chaves AES de 256 bits.
- A identificação e a proteção de chaves simétricas distribuídas são consistentes com o ANSI X9 TR-34 e o PCI PIN Anexo A1, exceto para algoritmos assimétricos e pontos fortes de chave apropriados para proteger chaves AES de 256 bits.

As chaves principais da região são estabelecidas para HSMs aquelas que foram autenticadas e provisionadas para uma região por:

- Uma chave principal é gerada em um HSM na região. Esse HSM é designado como o host de distribuição de chaves.
- Todos os provisionados HSMs na região geram o token de autenticação KRD, que contém a chave pública do HSM e informações de autenticação não reproduzíveis.
- Os tokens KRD são adicionados à lista de permissões do KDH depois que o KDH valida a identidade e a permissão do HSM para receber as chaves.
- O KDH produz um token de chave principal autenticável para cada HSM. Os tokens contêm informações de autenticação KDH e uma chave principal criptografada que pode ser carregada somente em um HSM para o qual ela foi criada.
- Cada HSM recebe o token de chave principal criado para ele. Depois de validar as informações de autenticação do próprio HSM e as informações de autenticação do KDH, a chave principal é descriptografada pela chave privada do KRD e carregada na chave principal.

Caso um único HSM precise ser sincronizado novamente com uma região:

- Ele é revalidado e provisionado com firmware e configuração.
- Se for novo na região:
  - O HSM gera um token de autenticação KRD.
  - O KDH adiciona o token à sua lista de permissões.
  - O KDH gera um token de chave principal para o HSM.
  - O HSM carrega a chave principal.
  - O HSM é disponibilizado para o serviço.

Isso garante que:

- Somente o HSM validado para processamento AWS de criptografia de pagamento em uma região pode receber a chave mestra dessa região.
- Somente uma chave mestra de um HSM de criptografia de AWS pagamento pode ser distribuída para um HSM da frota.

Alternância de chaves principais da região

As chaves principais da região são alternadas ao fim do período criptográfico, no caso improvável de uma suspeita de comprometimento da chave ou após alterações no serviço que possam afetar a segurança da chave.

Uma nova chave principal de região é gerada e distribuída, como no provisionamento inicial. As chaves principais do perfil salvas devem ser traduzidas para a nova chave principal da região.

A alternância da chave principal da região não afeta o processamento do cliente.

Sincronização da chave principal do perfil

As chaves principais do perfil são protegidas pelas chaves principais da região. Isso restringe um perfil a uma região específica.

As chaves principais do perfil são provisionadas adequadamente:

- Uma chave principal de perfil é gerada em um HSM que tem a chave principal da região sincronizada.
- A chave principal do perfil é armazenada e criptografada com a configuração do perfil e outros contextos.
- O perfil é usado para funções criptográficas do cliente por qualquer HSM na região com a chave principal da região.

Alternância de chaves principais do perfil

As chaves principais do perfil são alternadas ao fim do período criptográfico, após suspeita de comprometimento da chave ou após alterações no serviço que possam afetar a segurança da chave.

Etapas de alternância:

- Uma nova chave principal de perfil é gerada e distribuída como uma chave principal pendente, assim como no provisionamento inicial.

- Um processo em segundo plano converte o material de chave do cliente da chave principal do perfil estabelecido para a chave principal pendente.
- Quando todas as chaves do cliente tiverem sido criptografadas com a chave pendente, ela será promovida à chave principal do perfil.
- Um processo em segundo plano exclui o material de chave do cliente protegido pela chave expirada.

A alternância da chave principal do perfil não afeta o processamento do cliente.

Proteção

As chaves dependem somente da hierarquia de chaves para a proteção. A proteção das chaves principais é fundamental para evitar a perda ou o comprometimento de todas as chaves do cliente.

As chaves principais da região podem ser restauradas somente do backup para o HSM autenticado e provisionado para o serviço. Essas chaves podem ser armazenadas apenas como tokens de chave principal criptografados e mutuamente autenticáveis de um KDH específico para um HSM específico.

As chaves mestras do perfil são armazenadas com a configuração do perfil e as informações de contexto criptografadas por região.

As chaves do cliente são armazenadas em blocos de chaves, protegidas por uma chave mestra de perfil.

Todas as chaves existem exclusivamente em um HSM ou são armazenadas protegidas por outra chave de força criptográfica igual ou mais forte.

Durabilidade

As chaves do cliente para criptografia de transações e funções de negócios devem estar disponíveis mesmo em situações extremas que normalmente causariam interrupções. AWS A criptografia de pagamento utiliza um modelo de redundância de vários níveis em todas as zonas e regiões de disponibilidade. AWS Os clientes que precisam de maior disponibilidade e durabilidade para operações criptográficas de pagamento do que as fornecidas pelo serviço devem implementar arquiteturas multirregionais.

A autenticação do HSM e os tokens da chave principal são salvos e podem ser usados para restaurar uma chave principal ou sincronizar com uma nova chave principal, caso um HSM precise ser redefinido. Os tokens são arquivados e usados somente sob controle duplo quando necessário.

Acesso do operador às chaves principais do HSM

As chaves principais existem somente no HSM gerenciado pelo serviço e protegido em instalações seguras da AWS. As chaves principais não podem ser exportadas de nenhum HSM nem sincronizadas com um HSM que não seja inicializado pelo fabricante para uso no serviço. Os operadores da AWS não podem obter chaves principais em nenhum formato que possam ser carregadas em um HSM não gerenciado pelo serviço.

Gerenciamento de chaves de clientes

Na AWS, a confiança do cliente é nossa maior prioridade. Você mantém o controle total das chaves que você importa ou cria no serviço em sua conta da AWS. Você é responsável pela configuração do acesso às chaves.

O AWS Payment Cryptography é um provedor de serviços que usa HSMs e gerencia chaves em nome dos clientes, semelhante aos provedores de serviços de pagamento de longa data. O serviço tem total responsabilidade pela segurança física e lógica do HSM. A responsabilidade pelo gerenciamento de chaves é compartilhada entre o serviço e os clientes porque o cliente deve fornecer informações precisas sobre as chaves criadas ou importadas para o serviço, que o serviço usa para impor o uso e o gerenciamento corretos das chaves. As proteções de segregação de dados da AWS são usadas para garantir que o comprometimento das chaves pertencentes a uma conta da AWS não comprometa as chaves pertencentes a outra.

AWS A criptografia de pagamento tem total responsabilidade pela conformidade física do HSM e pelo gerenciamento de chaves das chaves gerenciadas pelo serviço. Isso requer a propriedade e o gerenciamento das chaves principais do HSM e a proteção das chaves do cliente gerenciadas pela criptografia AWS de pagamento.

Separação de espaço entre chaves do cliente

AWS A criptografia de pagamento aplica as principais políticas para todos os usos de chaves, incluindo a limitação dos diretores à conta proprietária da chave, a menos que uma chave seja explicitamente compartilhada com outra conta.

As contas da AWS fornecem segregação completa do ambiente entre clientes ou aplicativos, de forma análoga às implementações fora da nuvem em diferentes datacenters. Cada conta fornece controle de acesso isolado, redes, recursos computacionais, armazenamento de dados, chaves criptográficas para proteção de dados e transações de pagamento, além de todos os recursos da AWS. Os serviços da AWS, como Organizations e Control Tower, permitem o gerenciamento

corporativo de contas de aplicativos separadas, de forma análoga a gaiolas ou salas em um datacenter corporativo.

Acesso do operador às chaves do cliente

As chaves de cliente gerenciadas pelo serviço são armazenadas protegidas por chaves principais de partição e só podem ser usadas pela conta do cliente proprietário ou pela conta que o proprietário configurou especificamente para o compartilhamento de chaves. Os operadores da AWS não podem exportar ou realizar operações criptográficas ou de gerenciamento de chaves com chaves de clientes usando o acesso manual ao serviço, que é gerenciado pelos mecanismos de acesso manual do operador da AWS.

O código de serviço que implementa o gerenciamento e o uso de chaves de clientes está sujeito às práticas de código seguro da AWS, conforme avaliado pela avaliação do PCI DSS da AWS.

Backup e recuperação

As chaves e as principais informações armazenadas internamente pelo serviço para uma região são copiadas em arquivos criptografados pelo AWS. Os arquivos exigem controle duplo AWS para serem restaurados.

Blocos de chaves

Todas as chaves são armazenadas e processadas em blocos de chaves no formato ANSI X9.143.

As chaves podem ser importadas para o serviço a partir de criptogramas ou outros formatos de bloco de chaves suportados pelo ImportKey. Da mesma forma, as chaves podem ser exportadas, se forem exportáveis, para outros formatos de blocos de chaves ou criptogramas compatíveis com os perfis de exportação de chaves.

Uso de chaves

O uso da chave é restrito ao configurado KeyUsage pelo serviço. O serviço falhará em qualquer solicitação com uso inadequado de chaves, modo de uso ou algoritmo para a operação criptográfica solicitada.

Relações de troca de chaves

O PCI PIN Security e o PCI P2PE exigem que as organizações que compartilham chaves que criptografam PINs ou armazenam dados, incluindo as chaves de troca de chaves (KEK) usadas para

compartilhar essas chaves, não compartilhem as mesmas chaves com nenhuma outra organização. É uma prática recomendada que as chaves simétricas sejam compartilhadas entre apenas duas partes para uma única finalidade, inclusive dentro da mesma organização. Isso minimiza o impacto de suspeitas de comprometimento de chaves que forcem a substituição das chaves afetadas.

Mesmo os casos de negócios que exigem o compartilhamento de chaves entre mais de duas partes devem manter um número mínimo de partes.

AWS A criptografia de pagamento fornece etiquetas-chave que podem ser usadas para rastrear e impor o uso de chaves dentro desses requisitos.

Por exemplo, KEK e BDK para diferentes instalações de injeção de chaves podem ser identificados definindo um “KIF” = “POSStation” para todas as chaves compartilhadas com esse provedor de serviços. Outro exemplo seria marcar chaves compartilhadas com redes de pagamento com “Rede” = “PayCard”. As tags permitem que você crie controles de acesso e crie relatórios de auditoria para aplicar e demonstrar suas principais práticas de gerenciamento.

Exclusão de chaves

DeleteKey marca as chaves no banco de dados para exclusão após um período configurável pelo cliente. Após esse período, a chave é excluída irreversivelmente. Esse é um mecanismo de segurança para evitar a exclusão acidental ou maliciosa de uma chave. As teclas marcadas para exclusão não estão disponíveis para nenhuma ação, exceto RestoreKey.

As chaves excluídas permanecem nos backups do serviço por sete dias após a exclusão. Elas não são restauráveis durante esse período.

As chaves pertencentes a contas fechadas da AWS são marcadas para exclusão. Se a conta for reativada antes que o período de exclusão seja atingido, todas as chaves marcadas para exclusão serão restauradas, mas desativadas. Elas devem ser reativadas por você para serem usadas em operações criptográficas.

Segurança de comunicação

Externo

AWS Os endpoints da API de criptografia de pagamento atendem aos padrões de AWS segurança, incluindo TLS na versão 1.2 ou superior e Signature versão 4 para autenticação e integridade das solicitações.

As conexões TLS de entrada são encerradas em balanceadores de carga de rede e encaminhadas para manipuladores de API por meio de conexões TLS internas.

Interno

As comunicações internas entre os componentes do serviço e entre os componentes do serviço e outros serviços da AWS são protegidas por TLS usando criptografia robusta.

Os HSM estão em uma rede privada, não virtual, que pode ser acessada apenas por componentes de serviço. Todas as conexões entre o HSM e os componentes do serviço são protegidas com TLS mútuo (mTLS), igual ou superior ao TLS 1.2. Os certificados internos para TLS e mTLS são gerenciados pelo Amazon Certificate Manager usando uma AWS Private Certificate Authority. A rede interna VPCs e a rede HSM são monitoradas em busca de atividades inesperadas e alterações de configuração.

Registro em log e monitoramento

Os registros de serviços internos incluem:

- CloudTrail registros de chamadas de serviço da AWS feitas pelo serviço
- CloudWatch registros de ambos os eventos registrados diretamente nos CloudWatch registros ou eventos do HSM
- Arquivos de log do HSM e dos sistemas de serviço
- Arquivos de log

Todas as fontes de log monitoram e filtram informações confidenciais, inclusive sobre chaves. Os logs são revisados sistematicamente para garantir que não contenham informações confidenciais do cliente.

O acesso aos logs é restrito às pessoas necessárias para concluir as funções de trabalho.

Todos os logs são retidos de acordo com as políticas de retenção de registros da AWS.

Operações do cliente

AWS A criptografia de pagamento tem total responsabilidade pela conformidade física do HSM de acordo com os padrões PCI. O serviço também fornece um armazenamento seguro de chaves

e garante que as chaves só possam ser usadas para os fins permitidos pelos padrões PCI e especificados por você durante a criação ou importação. Você é responsável por configurar os principais atributos e acesso para aproveitar os recursos de segurança e conformidade do serviço.

Tópicos

- Gerar chaves
- Importar chaves
- Exportar chaves
- Delete an key
- Alternar chaves do

Gerar chaves

Ao criar chaves, você define os atributos que o serviço usa para impor o uso compatível da chave:

- Algoritmo e comprimento da chave
- Usage
- Disponibilidade e validade

Tags usadas para o controle de acesso por atributo (ABAC) são usadas para limitar as chaves para uso com parceiros ou aplicativos específicos e também devem ser definidas durante a criação. Inclua políticas para limitar as funções permitidas para excluir ou alterar tags.

Você deve garantir que as políticas que determinam as funções que podem usar e gerenciar a chave sejam definidas antes da criação da chave.

> **Note**
>
> As políticas do IAM nos CreateKey comandos podem ser usadas para impor e demonstrar o controle duplo para a geração de chaves.

Importar chaves

Ao importar chaves, os atributos para impor o uso compatível da chave são definidos pelo serviço usando as informações vinculadas criptograficamente no bloco de chaves. O mecanismo para definir

o contexto de chave fundamental é usar blocos de chave criados com o HSM de origem e protegidos por uma KEK compartilhada ou assimétrica. Isso se alinha aos requisitos do PCI PIN e preserva o uso, o algoritmo e a força da chave do aplicativo de origem.

Atributos importantes, tags e políticas de controle de acesso devem ser estabelecidos na importação, além das informações no bloco de chaves.

A importação de chaves usando criptogramas não transfere atributos de chave do aplicativo de origem. Você deve definir os atributos adequadamente usando esse mecanismo.

Frequentemente, as chaves são trocadas usando componentes de texto não criptografado, transmitidas pelos guardiões das chaves e, em seguida, carregadas com uma cerimônia que implementa o controle duplo em uma sala segura. Isso não é diretamente suportado pela criptografia AWS de pagamento. A API exportará uma chave pública com um certificado que pode ser importado pelo seu próprio HSM para exportar um bloco de chaves que pode ser importado pelo serviço. Isso permite o uso de seu próprio HSM para carregar componentes de texto não criptografado.

Você deve usar valores de verificação de chave (KCV) para verificar se as chaves importadas correspondem às chaves de origem.

As políticas do IAM na ImportKey API podem ser usadas para impor e demonstrar o controle duplo para a importação de chaves.

Exportar chaves

O compartilhamento de chaves com parceiros ou aplicativos on-premises pode exigir a exportação de chaves. O uso de blocos de chaves para exportações mantém o contexto fundamental da chave com o material de chave criptografado.

Tags de chave podem ser usadas para limitar a exportação de chaves para KEKs que compartilham a mesma tag e valor.

AWS A criptografia de pagamento não fornece nem exibe os principais componentes em texto não criptografado. Isso requer acesso direto dos principais guardiões aos dispositivos criptográficos seguros (SCD) testados pelo PCI PTS HSM ou ISO 13491 para exibição ou impressão. Você pode estabelecer uma KEK assimétrica ou simétrica com seu SCD para conduzir a cerimônia de criação de componentes de chave de texto não criptografado sob controle duplo.

Os valores de verificação de chave (KCV) devem ser usados para verificar se as chaves de origem importadas pelo HSM de destino correspondem às chaves de origem.

Delete an key

É possível usar a API DeleteKey para programar a exclusão das chaves após um período configurado por você. Antes disso, as chaves podem ser recuperadas. Depois que as chaves são excluídas, elas são removidas do serviço permanentemente.

As políticas do IAM na DeleteKey API podem ser usadas para impor e demonstrar o controle duplo para a exclusão de chaves.

Alternar chaves do

O efeito da alternância da chave pode ser implementado usando o alias da chave criando ou importando uma nova chave e modificando o alias da chave para se referir à nova chave. A chave antiga seria excluída ou desativada, dependendo de suas práticas de gerenciamento.

Cotas para AWS Payment Cryptography

Sua conta da AWS tem cotas padrão, anteriormente chamadas de limites, para cada serviço da AWS. A menos que especificado de outra forma, cada cota é específica da região. Você pode solicitar o aumento de algumas cotas, porém, algumas delas não podem ser aumentadas.

Nome	Padrão	Ajustá	Description
Aliases	Cada região com suporte: 2.000	Sim	O número máximo de alias que você pode ter nessa conta na Região atual.
Taxa combinada de solicitações de ambiente de gerenciamento	Cada região compatível: 5 por segundo	Sim	O número máximo de solicitações de ambiente de gerenciamento por segundo que você pode fazer nessa conta na região atual. Essa cota se aplica a todas as operações combinada s do ambiente de gerenciamento.
Taxa combinada de solicitações de planos de dados (assimétrica)	Cada região compatível: 20 por segundo	Sim	O número máximo de solicitações por segundo para as operações de plano de dados com chaves assimétricas que você pode fazer nessa conta na região atual. Essa cota se aplica a todas as operações combinadas de plano de dados.

Nome	Padrão	Ajustá	Description
Taxa combinada de solicitações de planos de dados (simétrica)	Cada região compatível: 500 por segundo	Sim	O número máximo de solicitações por segundo para as operações de plano de dados com chaves simétricas que você pode fazer nessa conta na região atual. Essa cota se aplica a todas as operações combinadas de plano de dados.
Chaves	Cada região com suporte: 2.000	Sim	O número máximo de chaves que você pode ter nessa conta na Região atual, exceto chaves de exclusão.

Histórico de documentos para o AWS Guia do usuário de criptografia de pagamento

A tabela a seguir descreve os lançamentos da documentação sobre criptografia AWS de pagamento.

Alteração	Descrição	Data
Suporte de chave assimétrica para Multi-Region replicação de chaves	Multi-Region a replicação de chaves agora oferece suporte a chaves de criptografia de pagamento assimétricas, como RSA e ECC, além de chaves simétricas.	27 de julho de 2026
Novo recurso - AS2805	Suporte para algoritmos e fluxos para suportar o suporte regional do AS2805	17 de dezembro de 2025
Novo recurso - replicação de Multi-Region chaves	Com a replicação de Multi-Reg ion chaves, você pode replicar suas chaves de criptografia AWS de pagamento para várias. Regiões da AWS	10 de setembro de 2025
Novo recurso - ECDH	Com esta versão, o ECDH pode ser usado para estabelec er um KEK compartilhado para posterior troca de chaves.	30 de março de 2025
Nova orientação sobre troca de chaves	Novas orientações fornecida s para trocas de chaves. Informações sobre comandos comuns do JCB também foram adicionadas.	31 de janeiro de 2025
Lançamento de uma nova região	Foram adicionados endpoints para o lançamento de novas	31 de julho de 2024

	regiões na Europa (Frankfurt), Europa (Irlanda), Ásia-Pacífico (Cingapura) e Ásia-Pacífico (Tóquio)	
CloudTrail para plano de dados e chaves dinâmicas	Foram adicionadas informações sobre a utilização CloudTrail para operações de plano de dados (criptográficas), incluindo exemplos. Também foram adicionadas informações sobre a utilização de chaves dinâmicas para determinadas funções para oferecer melhor suporte a chaves únicas ou de uso limitado que não devem ser importadas para AWS a criptografia de pagamento	10 de julho de 2024
Exemplos atualizados	Novos exemplos adicionados para emissão de cartões	1º de julho de 2024
Lançamento do recurso	Adicionar informações sobre endpoints de VPC (PrivateLink) e exemplos de iCVV.	30 de maio de 2024
Lançamento do recurso	Informações adicionadas sobre novos recursos relacionados ao import/export uso de chaves RSA e exportação de chaves IPEK/IK DUKPT.	15 de janeiro de 2024
Lançamento inicial	Versão inicial do Guia do usuário AWS de criptografia de pagamento	8 de junho de 2023

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.